Improved Algorithms for Hybrid Video Coding

Chia Woo Yu, BSc, MPhil.

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Computer Science

University of Warwick, Department of Computer Science
January 2007

Improved Algorithms for Hybrid Video Coding

Chia Woo Yu, BSc, MPhil.
A thesis submitted to
The University of Warwick
For the degree of Doctor of Philosophy
January 2007

Summary

The integration of digital audio-visual data has played a pivotal role in the commercial success of multimedia communication. Due to the volume of data it represents, raw digital materials require enormous storage capacity and high bandwidth requirements for transmission. Hybrid video coding has become the most widely used compression technique in video coding standards such as MPEG-x and H.26x. This thesis develops several improved algorithms for both block-based and mesh-based hybrid video coding. In the block-based system, the research focuses mainly on the design of efficient implementations for the H.264/AVC video coding standard. The thesis discusses several viable algorithms for intra-frame coding and inter-frame coding. The performance of the proposed algorithms has been examined extensively using test sequences featuring different spatial composition and motion activity. The simulation results indicate a similar visual quality to the H.264/AVC standard in all objective and subjective tests, yet achieve a saving in encoding time of up to 77%.

The thesis also considers mesh-based video coding. A dual-layer motion compensation technique is developed to provide an alternative to the conventional approach of using single fixed patch size. The dual-layer approach is developed to overcome the constraints of connectivity and coding overhead. Overall, the proposed dual-layer structure is shown to achieve significant improvements of up to 2dB over traditional mesh-based coding at fixed bit-rates.

Keywords: hybrid video coding, H.264/AVC, intra-frame coding, inter-frame coding, mesh-based motion estimation, efficient algorithm.

Contents

Abstract ii

Contents iii

List of Figures viii

List of Tables ix

Acknowledgements xiv

Declaration xv

Abbreviations xvi

Chapter 1 Introduction and Overview 1

1.1 Digital Video Coding 2

1.1.1 Digital Video Coding Standards 2

1.1.2 State-of-the-Art 5

1.2 Thesis Outline 6

Chapter 2 Hybrid Video Coding 8

2.1 Overview 8

2.2 Structure of Generic Video Codec 9

2.2.1 Spatial Prediction and Transform Coding 11

2.2.2 Motion Estimation and Motion Compensation 14

2.2.3 Entropy Coding 18

2.3 Structure of H.264/AVC Standard 20

2.3.1 Intra-Coding with Directional Extrapolation 20

2.3.2 Motion Estimation with Multiple Partitioned Blocks 26

2.3.3 Context-based Adaptive Coding 29

2.4 Performance Comparison 31

2.4.1. Test Material 31

2.4.2. Objective Quality Assessments 32

2.4.3. Performance Comparisons 33

2.5 Summary 36

Chapter 3 Frequency Domain Approach To Fast Intra-frame Coding 38

3.1 Literature Review 39

3.2 Proposed Frequency Domain Approach 41

3.2.1 Algorithm Formulation 42

3.2.2 Algorithm Description 47

3.3 Simulations, Comparisons, and Discussions. 49

3.3.1 Simulation Results for Various Values of *M* 49

3.3.2 Comparisons with Other Algorithms 53

3.4 Summary 57

Chapter 4 Multiple Mode Selection for Inter-frame Coding **59**

4.1 Existing Solutions for Fast Inter-frame Coding 60

4.1.1 Early Detection for Skipped Macroblocks 61

4.1.2 Fast Algorithms Featuring Hierarchical Decisions 62

4.2 The Proposed *Finter1* and *Finter2* Algorithms 63

4.2.1 Algorithm Formulation 63

4.2.2 The *Finter1* Algorithm 66

4.2.3 The *Finter2* Algorithm 70

4.3 Simulations, Comparisons, and Discussions 74

4.3.1 Picture Degradation, Compression Ratio, and Speedup 74

4.3.2 Rate-distortion Performance 77

4.4 Summary 81

Chapter 5 Improved Scheme for Multiple Mode Selection **82**

5.1 Improved Scheme for the *Finter2* Algorithm 83

5.1.1 Advanced Skipped Macroblock Detector 83

5.1.2 Coding for Homogeneous Macroblock 89

5.1.3 Mode Selection for High-detailed Macroblock 92

5.2 Simulations, Comparisons and Discussions 94

5.2.1 Performance Comparison for Inter-frame Coding 94

5.2.2 Performance Comparison for Hybrid Coding Structure 102

5.3 Summary 107

Chapter 6 **Mesh-based Motion Compensation Featuring a Dual Layer Structure** **109**

6.1 Overview 109

6.1.1 Mesh Size v. Rate-distortion Performance 111

6.1.2 Motion Coding v. Residue Coding 114

6.2 Dual Layer Structure for Mesh-based Video Coding 115

6.2.1 Estimation in the Basic Layer 116

6.2.2 Construction of the Significance Maps 117

6.2.3 Refinement in the Progressive layer 121

6.2.4 Group-wise Motion Coding 123

6.2.5 Decoder of the Proposed Algorithm 127

6.3 Simulations, Comparisons, and Discussions. 128

6.3.1 Prediction Accuracy and Motion Coding Bit Rate 128

6.3.2 Algorithm Complexity 130

6.3.3 Rate-Distortion Performance 131

6.4 Summary 133

Chapter 7 **Conclusions** **138**

7.1 Transform Domain Approach to Fast Intra-frame Coding 138

7.2 Multiple Mode Selection for H.264/AVC Inter-frame Coding 141

7.3 Mesh-based Motion Compensation Featuring a Dual-layer Structure 145

7.4 Further Work 147

7.5 Concluding Remarks 149

Appendix 1 Image Test Sequences **150**

Appendix 2 The coefficients of Ω_{mode} at various frequency positions **153**

Bibliography **155**

Publications **164**

List of Tables

2-1	Average bit-rate savings of H.264/AVC in comparison with other standards.	34
3-1	The average Y-PSNR difference, bit-rate difference, and time saving for the Fintra algorithms with different M settings.	51
3-2	The average time saving achieved by the *Fintra* algorithm and Pan et al's algorithm.	54
4-1	Average proportion of skipped macroblocks in inter-frame coding with respect to compression levels (indicated by Qp) and image motion activity (categorised by class).	61
4-2	Relationship between the three categories in the proposed algorithm and the 9 inter-predicted modes.	67
4-3	The Y-PSNR difference, bit-rate difference, and time saving of the *Finter1* and *Finter2* algorithms averaged for the different compression ratios.	76
5-1	Values of $T_{Qz,0}$ and $T_{Qz,1}$ with respect to Qp factor	87
5-2	Simulation results for three fast inter-mode selection algorithms, each compared with the H.264 JM6.1e software. (Except were stated, the sequences are of QCIF resolution).	96
5-3	Computational reduction compared to the JM6.1e encoder.	100
5-4	Computational reduction of combined algorithm compared to the JM6.1e encoder.	103
6-1	Average PSNR per frame prior to the residue coding.	132
6-2	Average Motion Bits per frame prior to the residue coding.	132
6-3	Average Complexity (affine iterations) per frame.	132
6-4	The conversion table for the two popular bit rate measurements for a QCIF picture decoded at 30Hz	131

List of Figures

1-1	The chronology of the video coding standards developed by ITU-T and MPEG.	3
2-1	Block structure of generic video codec: (a) encoder, (b) decoder.	10
2-2	Spatial prediction in Differential Pulse Code Modulation (DPCM).	11
2-3	(a) Motion estimation takes place between the current frame and the reference frame within a search window; (b) The search window and the definition of motion vector.	15
2-4	The positions of sub-pixel checking points (v, h, and c) surrounding an integer motion vector A.	17
2-5	DCT coefficient zigzag scan for an 8x8 block (non-interlaced frame).	19
2-6	Block structure of the H.264 codec: (a) encoder, (b) decoder.	21
2-7	(a) 4x4 block with elements (a to p) and neighbouring pixels (A to M); (b) Eight direction-biased intra-modes in a 4x4 block.	23
2-8	Example of *I4MB* prediction for an 4x4 block.	23
2-9	Two types of block decompositions for motion estimation: macroblock-type (top) and subblock-type (bottom).	27
2-10	Simplified block diagram of a Context-based Adaptive Binary Arithmetic Coding (CABAC) encoder.	30
2-11	Spatial distribution of luminance (Y) and chrominance (CrCb) components to form different formats for a colour picture.	32
2-12	Performance of four video coding standards, H.264/AVC, MPEG-4, H.263, and MPEG-2 for the *Foreman* (top) and *Tempete* (bottom) sequences.	35
3-1	Match percentage between the least distortion cost acquired from SAD implementation and the least rate-distortion cost obtained from Lagrangian evaluation.	41

3-2 The extrapolation scheme with diagonal down-left direction and its extrapolation equations for a 4x4 block. 46

3-3 The PSNR-rate distortion curves for the *Mobile & Calendar* sequence (Class C) obtained by the JM6.1e algorithm and the proposed *Fintra* algorithm with different *M* settings. 52

3-4 The PSNR-rate distortion curves for the *Stefan* sequence (Class C) obtained by the JM6.1e algorithm and the proposed *Fintra* algorithm with different *M* settings. 52

3-5 Performance comparison of the proposed *Fintra* algorithm and Pan et al's algorithm employing *Mobile & Calendar* sequence (Class C) in terms of rate-distortion (above); Magnified section of the curves (below). 55

3-6 Performance comparison of the proposed *Fintra* algorithm and Pan et al's algorithm employing *Stefan* sequence (Class C) in terms of rate-distortion (above); Magnified section of the curves (below). 56

4-1 Proposed scanning order for E_n and S_n, the energy and sum of intensities in a 4×4 block in order to reduce computational redundancy. 65

4-2 Flowchart of the proposed *Finter1* algorithm incorporating the complexity measurement for a macroblock. A macroblock will be recognized as one of three categories, namely *Cat0*, *Cat1* and *Cat2*. 69

4-3 The relative position of four nearest encoded neighbours with respect to the currently processed macroblock. 71

4-4 Flowchart of the proposed *Finter2* algorithm incorporating the complexity measurement for a macroblock, temporal similarity, and the detection of different moving features within a macroblock. 73

4-5 PSNR-rate distortion curves for the *Container* (Class A) obtained by the JM6.1e algorithm and the proposed *Finter1* and *Finter2* algorithms. 78

4-6 PSNR-rate distortion curves for the *Paris* (Class B) obtained by the JM6.1e algorithm and the proposed *Finter1* and *Finter2* algorithms. 78

4-7 PSNR-rate distortion curves for the *Mobile & Calendar* (Class C) obtained by the JM6.1e algorithm and the proposed *Finter1* and *Finter2* algorithms. 79

4-8 PSNR-rate distortion curves for the *Harbour* (Class C) obtained by the JM6.1e algorithm and the proposed *Finter1* and *Finter2* algorithms. 79

4-9 PSNR-rate distortion curves for the *FunFair* (Class C) obtained by the JM6.1e algorithm and the proposed *Finter1* and *Finter2* algorithms. 80

4-10 PSNR-rate distortion curves for the *Stefan* (Class C) obtained by the JM6.1e algorithm and the proposed *Finter1* and *Finter2* algorithms. 80

5-1 Flowchart of the first level of improved scheme that targets macroblocks encoded by *SKIP* mode. 84

5-2 Neighbouring macroblocks located in the current frame and the reference frame used for computation of Thd_A. 85

5-3 Skip-and-pick strategy to reduce time for examination of transform-quantised coefficients in a macroblock. 88

5-4 Flowchart of the second level that targets macroblocks encoded by inter modes with large partition size. 90

5-5 Flowchart of the third level that considers macroblocks encoded with *P8x8* mode. 93

5-6 PSNR-rate relationship diagram for the *Container* (Class A) sequence. 97

5-7 PSNR-rate relationship diagram for the *Paris* (Class B) sequence. 97

5-8 PSNR-rate relationship diagram for the *Mobile and Calendar* (Class C) sequence. 98

5-9 PSNR-rate relationship diagram for the *Harbour* (Class C) sequence. 98

5-10 PSNR-rate relationship diagram for the *Fun Fair* (Class C) sequence. 99

5-11 PSNR-rate relationship diagram for the *Stefan* (Class C) sequence. 99

5-12 Magnified snapshots from frame #9 of the *Stefan* sequence with *Qp*=28. (Top) encoded by JM6.1e; the *Finter2* method; (Bottom) Wu et al's algorithm; the proposed scheme. 101

5-13 PSNR-rate relationship diagram for the *Container* (Class A) sequence. 104

5-14 PSNR-rate relationship diagram for the *Paris* (Class B) sequence. 104

5-15 PSNR-rate relationship diagram for the *Mobile and Calendar* (Class C) sequence. 105

5-16 PSNR-rate relationship diagram for the *Harbour* (Class C) sequence. 105

5-17 PSNR-rate relationship diagram for the *Fun Fair* (Class C) sequence. 106

5-18 PSNR-rate relationship diagram for the *Stefan* (Class C) sequence. 106

6-1 *Football* sequence predicted by deformed triangular mesh of size 16x16. 110

6-2 Rate-distortion performance of *Football* sequence employing mesh-coding with different patch sizes. 112

6-3 Transitional conversion process from reference frame, F_{t-1}, to target frame, F_t;. 113

6-4 PSNR improvement and bit allocation for the Football sequence coded at 0.30 bits/pixel. 113

6-5 Proposed conversion process employing two motion estimation processes prior to residue coding. 113

6-6 Motion estimation is elaborated to three stages followed by group-wise entropy coding. The shaded boxes represent new features introduced by the proposed dual-layer structure. 116

6-7 Search Scheme for the hexagon matching algorithm: (Left) Fix the position of the six vertices, G_a to G_f ; (Right) Search for best position of G_z. 117

6-8 Block diagram showing the construction of significance maps, where M_C and M_R are the coarse and refined maps, respectively. 118

6-9 Effective area for modified MAD evaluation in Equation (6-1) and (6-3). 119

6-10 The precision conversion between 16x16 and 8x8 grid point coordinate systems. 122

6-11 Proposed entropy coding for motion information. (a) Conventional approach using raster scan; (b) proposed approach incorporating indicator bits and a group-wise scan resulting in fewer bits to encode the same amount of motion information. 123

6-12 Available pattern descriptions for motion plane in the progressive layer. The top four models (H1 ~ H4) represent the 'header' of an isolated motion-activity region, followed by subsequent models (S1 ~ S3). 124

6-13 The application of pattern description on motion plane of irregular size in the progressive layer. 125

6-14 The diagram of the proposed decoder for the dual-layer structure. 127

6-15 PSNR-rate diagrams for *Hall Monitor* sequence in Class A. 135

6-16 PSNR-rate diagrams for *Silent Voice* sequence in Class B. 135

6-17 PSNR-rate diagrams for *Fun Fair* sequence in Class C. 136

6-18 PSNR-rate diagrams for *Ice Skating* sequence in Class C. 136

6-19 PSNR-rate diagrams for *Mobile and Calendar sequence* in Class C. 137

6-20 PSNR-rate diagrams for *Stefan* sequence in Class C. 137

7-1 General block structure of the scalable video coding technique. 147

Acknowledgements

I would like to take this opportunity to show my appreciation to my supervisor Graham Martin who constantly offers the guidance and support. I am particularly grateful to his prompt response in the thesis poof-reading.

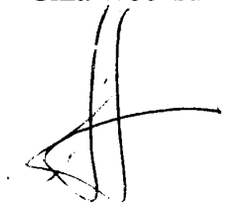
Thanks are due to my colleague, Heechan Park, for his discussions and inspirations in the research work. During the two years of collaboration, a fruitful achievement in publication is in evidence.

Thanks are deserved to my best friends, Alisa Săkič and Donna Lee-Edney, to provide distractions during the painstaking writing process. Special thanks must go to my partner, Romanos Daskalou, for making my life more fulfilled. The unwavering encouragements came from him is the main support to finish the thesis.

Declaration

I declare that, except where acknowledged, the material contained in this thesis is my own work and that is has neither been previously published nor submitted elsewhere for the purpose of obtaining an academic degree.

Chia Woo Yu



Abbreviations

1D	One Dimension
2D	Two Dimensions
AC	Alternating Current (high frequency)
AV	Audio-Visual
AVC	Advanced Video Coding
BMA	Block Matching Algorithm
CABAC	Context-based Adaptive Binary Arithmetic Coding
CAVLC	Context Adaptive Variable Length Coding
CD	Compact Disk
CIF	Common Intermediate Format (352 x 288)
Codec	COder/DECoder
dB	DeciBel
DC	Direct Current (lowest frequency)
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DPCM	Differential Pulse Code Modulation
DST	Discrete Sine Transform
DVD	Digital Versatile Disc
EOF	End Of File
Finter1	Fast INTER mode selection algorithm 1
Finter2	Fast INTER mode selection algorithm 2
Fintra	Fast INTRA mode selection algorithm
FLC	Fix Length Coder/Coding
FLD	Fix Length Decoder/Decoding
FRext	Fidelity Range EXTension
HDTV	High Definition Television
HMA	Hexagonal Matching Algorithm
HORT	HORizonTal mode
IEC	International Electrotechnical Commision
IFinter	Improved Fast INTER mode selection algorithm
IntDCT	Integer Discrete Cosine Transform
Inv DCT	Inverse Discrete Cosine Transform
ISO	International Standard Organisation
ITU-T	International Telecommunication Union
JM	Joint Model
JTC	Joint Technical Committee
JVT	Joint Video Team
KLT	Karhunen-Loève Transform
LSI	Large-Scale Integration
MC	Motion Compensation
ME	Motion Estimation

MPEG	Moving Picture Experts Group
MPM	Most Probable Mode
MSE	Mean Square Error
MV	Motion Vector
MVD	Motion Vector Difference
NAL	Network Abstraction Layer
PMV	Predicted Motion Vector
PSNR	Peak Signal to Noise Ratio
PSTN	Public Switched Telephone Network
RD	Rate-Distortion
RDO	Rate-Distortion Optimisation
RGB	Red, Green, and Blue representations
QCIF	Quarter Common Intermediate Format (176 x 144)
SAD	Sum of Absolute Difference
SAE	Sum of Absolute Error
VCEG	Video Coding Experts Group
VCL	Video Coding Layer
VCR	Video Cassette Recorder
VERT	VERTical mode
VLC	Variable Length Coder/Coding
VLD	Variable Length Decoder/Decoding
VLSI	Very-Large-Scale Integration

Chapter 1

Introduction and Overview

During the last few decades, there has been a conversion from analogue to digital techniques for the storage and processing of audio-visual (AV) in both television broadcasting and recording systems. The systems integrated with digital information have also led to commercial success in the worlds of mobile telecommunications and digital entertainment. Raw digital video data require enormous storage capacity and significantly large bandwidth for transmission. This has sparked great interest in the development of compression techniques to reduce the amount of digital data without losing the representation of the source material. In the past few years, the video compression technologies have been implemented in VLSI hardware, resulting in significant inflexibility [1]. With advances in computer technology, software compression techniques have become viable. This has created a new dimension to the study of alternative solutions for digital video coding.

1.1 Digital Video Coding

Digital video has two coding formats: video file format (used for storage) and streaming file format (for transmission). Apart from a slight difference in header information, both formats share a similar goal: removing redundant information while maintaining visually acceptable quality. In theory, compression of a two-dimensional video signal can be achieved by exploiting spatial correlation [3]. Successive video frames also exhibit temporal redundancy, as groups of pixels remain the same from frame to frame [4, 5, 6].

Early video coding techniques, for example, differential pulse code modulation (DPCM) (described in Section 2.2.1) worked on a pixel-by-pixel basis to achieve compression. DPCM manifests good spatial resolution, but at the cost of an increased bandwidth requirement. It was soon realised that fractional bits could be employed to represent each pixel, permitting a group of samples to be coded together. The most straightforward way to do this was to deal with a collection/block of, say, 16x16 pixels [7]. This led to the block-based coding concept which has become the blueprint for generic video codecs (*co*ders/ *dec*oders).

1.1.1 Digital Video Coding Standards

The advance of digital video coding has entailed the convergence of sophisticated techniques and/or new features to satisfy the demands of new applications. Standardisation has played a significant role in the design of efficient video codecs. The Video Coding Experts Group (VCEG) in the International Telecommunication Union (ITU-T) and the Moving Picture Experts Group (MPEG) in the International Standard Organisation & International Electrotechnical Commission, Joint Technical Committee number one (ISO/IEC JTC1) are the leading organisations in the development of video

coding standards. The difference is that the former contributes efforts for telecommunication applications, such as video conferencing and video telephony, whereas MPEG mostly addresses the needs for video storage, broadcasting, and video streaming. The following paragraphs summarise the chronological development of video coding standards (shown as in Fig 1-1):

Early video coding standardisation efforts focussed mainly on applications for video conferencing. More specifically, the success of ITU-T H.261 [8] was a milestone for video coding at ISDN bit-rates (multiples of 64Kbits/*s*) based on the Common Intermediate Format (CIF) resolution of 352 x 288 pixels. Later, ISO/IEC JTC1 started developing a new coding technique, MPEG-1 [9]. The aim of MPEG-1 codec was to encode feature length films for storage on one or two digital Compact Disks (CD) at bit-rates of up to 1.5Mbits/s, with a performance comparable to that of VHS home video cassette recorders (VCRs). Since coding delays were considered less important in the recording process, motion compensation techniques (described in Section 2.2.2) were included to remove the temporal correlation between successive frames.

The digital revolution brought on by the MPEG-1 standard prompted the increasing demand for compression of audio-visual data at better resolutions. Two years

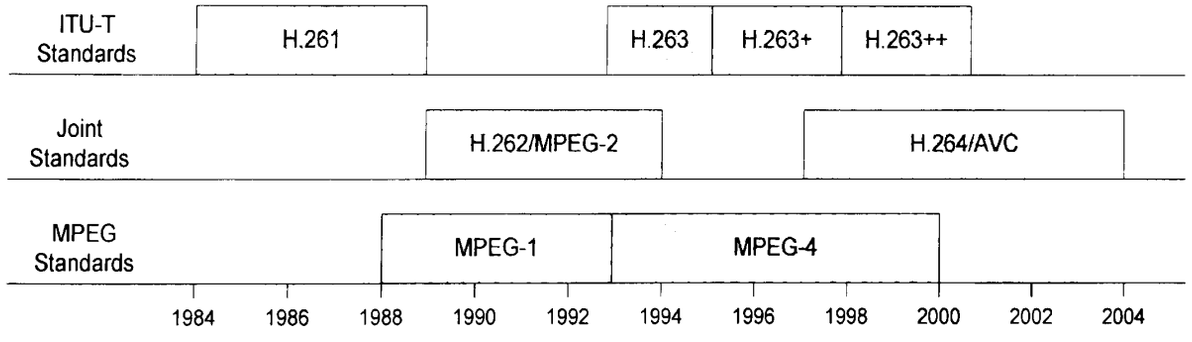


Fig. 1-1 The chronology of video coding standards developed by ITU-T and MPEG (diagram taken from [1]).

later, a new generation of MPEG coding, MPEG-2 [10], was introduced for video signals operating at higher bit-rates (4-9 Mbits/s) for full (interlaced) TV resolutions. This technique was also accepted for the ITU-T telecommunication systems under the generic name of H.262 [11]. The MPEG-2/H.262 standard made a significant impact in a range of applications, for example, storage onto digital versatile disc (DVD) and TV broadcasting over satellite, cable, and terrestrial systems. [1, 2].

Following the MPEG-2/H.262 standard, a requirement for sending video data at very low bit-rates arose from the advent of video transmission in public switched telephone networks (PSTN) and mobile applications. Consequently, VCEG established a new standard, namely, H.263 [12]. The H.263 standard fulfilled the goal to deliver video signals at bit-rates of 10-64 Kbits/s. The codec evolved over a number of years and was renamed H.263+ [13] and H.263++ [14] to indicate respective improvements. As a result of the efficient coding designs, the application of H.263 family products even spun off to very high resolution images, such as HDTV.

More recently, the MPEG-4 [15] standard was developed within an "object-based" framework [16] which allowed the images to be coded separately as individual objects. Object-based representations provide new functionalities, such as video interaction (objects can be accessed independently) and the virtual studio (permitting mixing of natural images with synthetic video). In terms of coding for frame-based natural images, the performance is quite similar to that of H.263.

1.1.2 State-of-the-Art

The MPEG-x and H.26x video coding standards have provided interoperability in heterogeneous network systems [20]. Considering that transmission bandwidth is still a valuable commodity, on-going developments in video coding seek to achieve additional compression while maintaining a reasonable level of signal-to-noise ratio. To this degree, VCEG embarked on a project in early 1998 with the aim of obtaining significant improvements over all existing coding methods. The project was named H.26L, with 'L' standing for long term objective of achieving this plan. A Joint Video Team (JVT) was established in 2001 to work on the ITU-T H.264 standard, also adopted by ISO/IEC JTC1 as MPEG-4 part 10, Advanced Video Coding (AVC).

The achievement of the H.264/AVC standard symbolises a major step forward in the development of modern video compression techniques. Due to its efficient design, the standard achieved a bit-rate saving of up to 50 percent compared to the most optimised H.263 or MPEG-4 codec at any compression level [17, 18, 19]. In other words, H.264/AVC provides a significant visual improvement for digital video content when operating at existing bandwidths. The H.264/AVC design includes the definitions of Video Coding Layer (VCL) and Network Abstraction Layer (NAL). The VCL covers the technical descriptions of the video compression, while NAL defines the format of the header information for transmission and storage. A further description of the H.264/AVC codec is given in the next chapter.

In the past, ITU-T and ISO/IEC JTC1 SC29 WG11 had little interest in common due to their distinct definitions for applications in the telecommunication and broadcasting fields, respectively. However in recent years, the border between the two fields has become less clear and a successful collaboration has resulted in a more sophisticated encoder design. The purpose of this thesis is to develop several improved

algorithms for techniques employed in the H.264/AVC standard. The ultimate goal is to reduce the complexity requirement of the encoders, without sacrificing the picture quality and compression ratio.

1.2 Thesis Outline

The next chapter contains a review of relevant research. Work done during the three-year PhD period is then detailed in the several methodology chapters. These include contributions made to techniques employed in the H.264/AVC standard (in Chapter 3, 4, and 5) and a mesh-based video codec (Chapter 6). Finally, conclusions are drawn in the final chapter. The detailed arrangements for each chapter are outlined as follows:

- Chapter 2 reviews the structure of generic hybrid coding for video signals, including intra-frame coding, inter-frame coding, and entropy coding techniques. Subsequently, advanced features integrated in the H.264/AVC standard are described to provide the knowledge required for the remaining chapters.

- Chapter 3 presents an intra-frame coding algorithm operating in the frequency domain. A literature review is also recorded prior to the algorithm formulation.

- Chapter 4 proposes two algorithms for fast mode selection in H.264/AVC inter-frame coding. Both algorithms are described as hierarchical structures comprising multiple levels. A performance comparison between the two algorithms is made.

- Chapter 5 contributes improvements to the algorithms described in the previous chapter. The work focuses on the overall performance of the algorithm, for instance, rate-distortion performance and speedup. In the simulation section, an extensive

experiment for both objective and subjective evaluation is described, and the improved algorithm is compared with other state-of-the-art methods.

- Chapter 6 presents a different coding structure based on a mesh-topology. An algorithm featuring two layers is proposed to improve coding efficiency. The proposed structure also includes the use of significance maps and group-wise coding to increase the performance of the encoder. The simulation results show prediction accuracy, compression efficiency, algorithm complexity, and rate-distortion performance.
- Finally, Chapter 7 provides some overall conclusions and directions for further work.

Chapter 2

Hybrid Video Coding

2.1 Overview

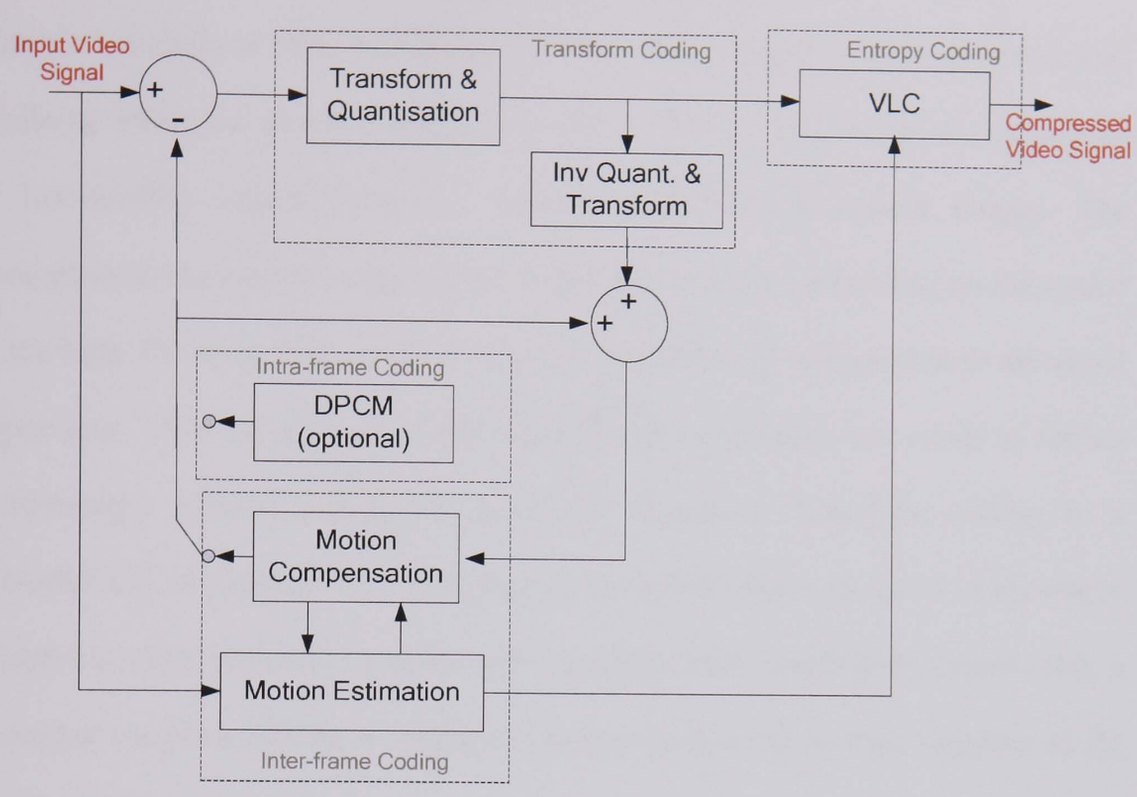
This chapter discusses the composite elements of both a generic video codec and the H.264/AVC coding technique. Similar to its predecessors, the H.264/AVC encoder is described as comprising a hybrid structure of intra-frame coding, inter-frame coding, and entropy coding. The intra-frame coding and inter-frame coding achieve compression by removing spatial and temporal dependencies that exist in the video source. In spite of working in different domains, both coding approaches involve transformation and quantisation processes. Subsequent to this, entropy coding is employed to assign pre-defined codewords representing the data, to reduce statistical redundancies. In the next two sections, the differences in hybrid structures of the generic codec and the H.264/AVC standard are described. A performance comparison of the two video codecs is made prior to the summary in Section 2.5.

2.2 Structure of Generic Video Codec

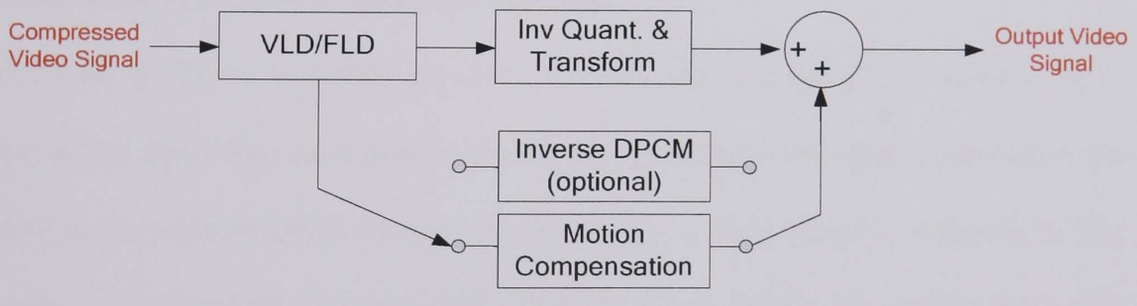
Throughout this thesis, the generic video codec referred to covers the MPEG-2/H.262, H.263 family and MPEG-4 video coding standards. Fig 2-1 illustrates the simplified block structures of the generic encoder and decoder. The encoder can be summarised as comprising three coding methods:

- *Intra-frame coding*: a strategy based on differential pulse code modulation (DPCM) is optionally employed to exploit the spatial dependencies within a frame. The residues of DPCM are further encoded by *transform coding* on the basis of 8x8 pixel blocks.
- *Inter-frame coding*: compression in a temporal fashion is achieved by block matching between successive frames. The prediction errors are encoded and eventually restored by motion compensation.
- *Entropy coding*: Variable Length Coding (VLC), i.e., Huffman coding or Arithmetic coding, is used to assign variable length codewords for data according to its frequency of occurrence.

In contrast, the process to decode a video frame is far more straightforward. It necessitates the inverse function of both VLC and transform coding to obtain the residue data. A reconstructed video frame is then restored by the residue data (if intra-coded), or by motion compensation (for inter-frame coding).



(a)



(b)

Fig. 2-1 Block structure of generic video codec: (a) encoder, (b) decoder.

2.2.1 Spatial Prediction and Transform Coding

Statistical analysis of video signals shows a significant proportion of local correlation/ similarity within the picture elements themselves. This is attributed to the prevalence of low-detailed content (such as smooth background) in natural images. The homogeneous regions may exhibit linear luminance variation depending on the source of the light. However, most of these intensity variations are unimportant to the visual appearance. Thus the loss of certain insignificant information can result in further compression, without introducing perceived distortion. Transform coding is a powerful tool to facilitate this compression by representing a group of video source samples (usually of size 8x8 pixels) with a few frequency coefficients. Prior to this, a technique based on DPCM is optionally used to reduce the average intensity of the pixel samples.

Differential Pulse Code Modulation (DPCM)

DPCM [3, 4, 22] is a simple method to reduce the average pixel intensity. It is achieved by predicting each pixel with one or more previously encoded samples, for example, the current pixel (marked by X in Fig. 2-2) is directly replaced by the sample C. Alternatively, a more sophisticated way is to predict the current pixel with the mean of the neighbouring samples, A, B, and C. The difference/residue between

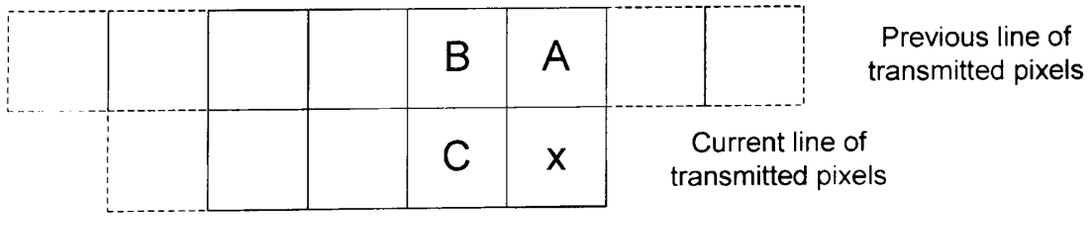


Fig. 2-2 Spatial prediction in Differential Pulse Code Modulation (DPCM).

the actual value and its predicted value is used for coding and transmission. Due to the property of spatial correlation, the intensity of the residue is usually smaller than the absolute value.

Transform Coding

Transform coding functions as a mapping process for a group of pixel samples to be represented in a more efficient way. This is attributed to the prevalence of homogeneous content in natural images. After the transformation, the pixel samples are represented by the same number of transform coefficients, but in the frequency domain rather than the spatial domain. The precision of each coefficient is reduced by a quantisation process. This provides compression. The precision of the low frequency coefficients is mainly retained due to the sensitivity of the human visual system to the low-detailed information.

Discrete Cosine Transform

The Two-Dimensional Discrete Cosine Transform (2D-DCT) [23] is used in most of the video coding standards. It is based on the principle that the spatial composition can be adequately represented by the linear combination of cosine waves at different frequencies. Let us denote $w(m,n)$ as a pixel sample at position (m, n) in an M-by-N block. After DCT, the same number of transform coefficients, $W(u,v)$, are obtained. The forward DCT is defined as:

$$W(u,v) = c(u)c(v)\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}\left[w(m,n)\cos\left(\frac{(2m+1)u\pi}{2M}\right)\cos\left(\frac{(2n+1)v\pi}{2N}\right)\right], \quad (2\text{-}1)$$

$$\text{where } c(u), c(v) = \begin{cases} \sqrt{\frac{1}{M}}, \sqrt{\frac{1}{N}} & for\ u,v = 0 \\ \sqrt{\frac{2}{M}}, \sqrt{\frac{2}{N}} & for\ u,v \neq 0 \end{cases} \quad \text{and } 0 \leq u < M \text{ and } 0 \leq v < N$$

The Inverse DCT (Inv DCT) is given by:

$$w(m,n)=\frac{2}{MN}\sum_{u=0}^{N-1}\sum_{v=0}^{M-1}\left[W(u,v)\cos\left(\frac{(2m+1)u\pi}{2M}\right)\cos\left(\frac{(2n+1)v\pi}{2N}\right)\right] \qquad (2\text{-}2)$$

Due to its separable property, the actual implementation for a forward 2D-DCT and its inverse is performed as a series of N 1D-DCTs in a horizontal manner, followed with M in the vertical direction.

The DCT has obtained wide acceptance for the following reasons: (a) the cosine kernels of the DCT possess smooth-varying bases which resemble the intensity variation of most natural images; (b) an identical compression gain to the Karhunen-Loève Transform (KLT) is achieved [24, 25]; (c) the availability of the fast DCT [26] provides an efficient implementation for both software-based video coding applications and VLSI designs [27].

Quantisation of DCT Coefficients

The mapping of the source signal into the DCT domain does not actually yield any compression due to the orthonormality property, i.e., no energy loss is incurred during the transformation process. Lossy coding can be achieved by a quantisation process which serves to reduce the precision of the DCT coefficients. The purpose of quantisation is to remove the frequency components that are relatively unimportant as far as visual appearance is concerned. Quantisation for a DCT coefficient, $W(u,v)$, is achieved by division by a quantisation value, $Q(u,v)$ and a rounding function:

$$W_q(u,v)=\text{round}\left(\frac{W(u,v)}{Q(u,v)}\right) \qquad (2\text{-}3)$$

Inverse quantisation can be performed by rescaling the quantised coefficients $W_q(u,v)$ by the quantisation value:

$$\hat{W}(u,v)=W_q(u,v).Q(u,v) \qquad (2\text{-}4)$$

The quantisation values can be either uniform (as in the H.263 family) or non-uniform (e.g., as in MPEG-4). In the former case, all the frequency coefficients are treated equally with a uniform quantisation value. In contrast, the latter strategy forms a quantisation matrix, Q, which has the same dimensions as that of the DCT block. The step of each quantisation coefficient (8-by-8 case) is determined according to the location where it resides:

$$\mathbf{Q}_{8\times 8} = \begin{bmatrix} 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 \\ 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 \\ 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 \\ 19 & 20 & 21 & 22 & 23 & 24 & 26 & 27 \\ 20 & 21 & 22 & 23 & 25 & 26 & 27 & 28 \\ 21 & 22 & 23 & 24 & 26 & 27 & 28 & 30 \\ 22 & 23 & 24 & 26 & 27 & 28 & 30 & 31 \\ 23 & 24 & 25 & 27 & 28 & 30 & 31 & 33 \end{bmatrix} \tag{2-5}$$

Due to the sensitivity of the human visual system, the low frequency locations have a lower quantisation step than those at higher frequencies.

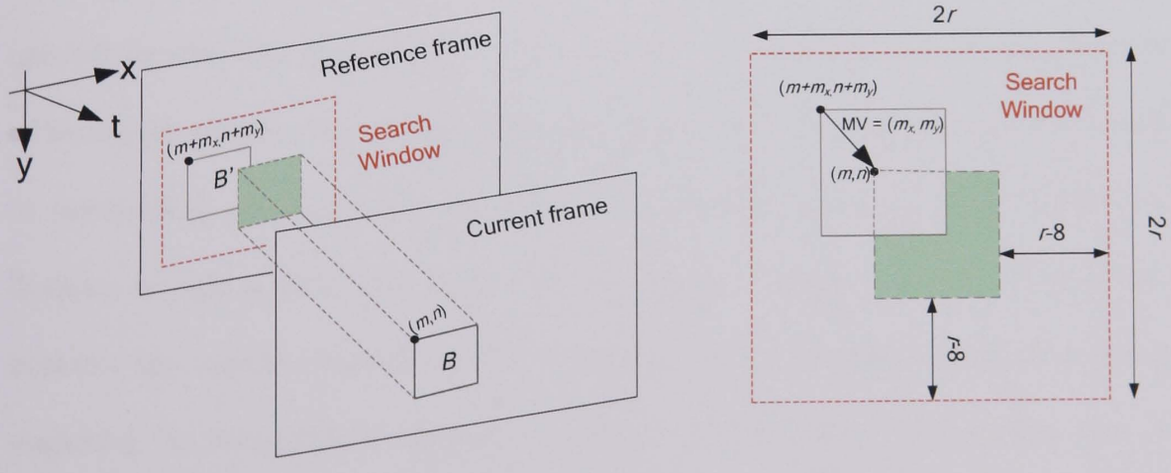
2.2.2 Motion Estimation and Motion Compensation

A video clip is actually a construction of images/pictures displayed in a sequential order. Thus, one expects a significant proportion of stationary information, such as static background to be present in successive frames. This temporal redundancy can be exploited by simply considering the difference/residue between successive frames. Moreover, a search strategy to approximate the trajectory of moving object using block-based translational movement between successive frames can significantly reduce the overall intensity of the residue data. The prediction of such translational movement is described by a pair of motion vectors.

Block-based Motion Estimation

The Block Matching Algorithm (BMA) [46] is the motion estimation technique currently adopted in all video coding standards. Initially, the frame is decomposed into blocks, usually 16-by-16 pixels. These are called macroblocks. Each macroblock could be coded differently as an intra-block or an inter-block. If inter-coding is chosen, a search is performed in a previously encoded frame within a search area of $2r$ x $2r$ pixels, i.e., the size of searching window, as shown in Fig. 2-3(b). The goal is to find the matching block that provides maximum correlation with the current macroblock. Various correlation measurements have been developed. Error metrics based on the Mean Square Error (MSE) [21] and the Sum of Absolute Difference (SAD) [28] are widely accepted due to a better compromise between good performance and simple computation. The MSE is defined as:

$$\mathrm{MSE}=\frac{1}{256}\sum_{m=0}^{15}\sum_{n=0}^{15}\left(B(m,n)-B'(m+m_x,n+m_y)\right)^2, \qquad |m_x,\ m_y| \le r\text{-}8 \qquad (2\text{-}6)$$



(a) (b)

Fig. 2-3 (a) Motion estimation takes place between the current frame and the reference frame within a search window; (b) The search window and the definition of 'motion vector'.

and SAD is defined as:

$$\text{SAD}=\sum_{m=0}^{15}\sum_{n=0}^{15}\left|B(m,n)-B'(m+m_x,n+m_y)\right|, \qquad \left|m_x, m_y\right| \le r\text{-}8 \qquad (2\text{-}7)$$

where $B(m,n)$ is the current macroblock with its top-left corner at coordinates (m,n), and $B'(m+m_x,\ n+m_y)$ is the corresponding block in the reference frame at new coordinates $(m+m_x,\ n+\ m_y)$, as indicated in Fig. 2-3(b).

After the search process, the position of the best matching block is described by a motion vector (MV). The MV is defined as the displacement of the best-matching block and the current block (Fig. 2-3(b)). The MV is the only information that needs to be coded and transmitted. Details of MV coding are described further in Section 2.2.3.

Motion Vectors with Half-Pixel Precision

Motion vectors with half-pixel resolution [45] can further improve the prediction accuracy. Without a significant increase in computation, a sub-pixel search is usually carried out after first obtaining an integer motion vector. In Fig. 2-4, A is the position of an integer motion vector found from a search process. The eight positions (v, h, and c) surrounding position A are the new checking points, each of which represent a distance of half a pixel. The search for the sub-pixel points requires interpolation between the sample elements in the reference frame. For each sub-pixel position marked h, the interpolation takes place between two horizontal integer pixels as $(A + X)/2$. Similarly, the interpolated value for position v is $(A + Y)/2$. As for the corner sub-pixels (centre of four integer pixels), the computation becomes $(A + X + Y + Z)/4$. Note that the divisions above are truncated to avoid fractional values.

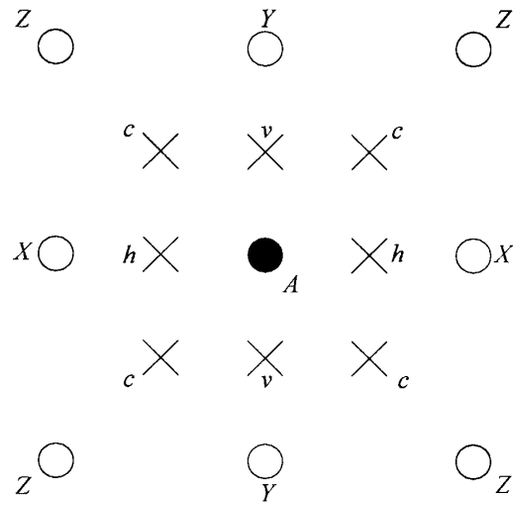


Fig. 2-4 The positions of sub-pixel checking points (v, h, and c) surrounding the integer motion vector A (diagram taken from [1]).

Motion Compensation

As indicated, the BMA considers only two-dimensional translational movement. One expects a significant prediction error to arise from other motion trajectories, such as, scaling and rotation. A prediction strategy based on image-warping may cope with this situation well, but at the cost of expensive computation (further consideration is given in Chapter 6). Furthermore, the occurrence of new scenes or objects in the current frame contributes to the difficulties in achieving an accurate prediction, since it is not possible to estimate them from previously encoded samples. Thus, motion compensation (MC) is employed to restore the loss of information caused by incorrect prediction. MC utilises transform coding, as described in Section 2.2.1. As the residue data rather than actual pixel values are coded, fewer bits are required. After transmission, the reconstructed residue data is restored by inverse quantisation and transformation and used for 'compensation' of the predicted block obtained from the reference frame.

2.2.3 Entropy Coding

After transform coding, each block is quantised, resulting in a few non-zero coefficients (significant coefficients). A zigzag scan is performed to reorder the block elements as a one-dimensional array. The statistical redundancies within the quantised coefficients (so-called symbols) can be removed by variable length coding (VLC), i.e., short codewords are allocated to frequent symbols, and vice versa.

Zigzag Scanning

Spatial reordering for a DCT block is essential for the efficiency of the entropy coding process. The aim is to rearrange all significant coefficients to form a compact representation. Scanning in a zigzag manner copes with the situation well. As illustrated in Fig. 2-5, the scan starts from the top-left corner where the lowest frequency component (DC) resides, through a zigzag track, and finishes at the highest frequency component at bottom-right. Note that the output of the zigzag reordering does not necessarily yield 64 coefficients, since zero coefficients are not required for coding. Thus, the last non-zero coefficient in the zigzag order is assigned a special symbol to specify the end of the file (EOF).

Variable Length Coding

After reordering, the significant coefficients in the DCT block are described in a compact 1D form. Variable length coding (VLC) is then employed to allocate different lengths of codeword according to the probability of occurrence, i.e., the more frequently occurring the symbol, the shorter the codeword. Huffman coding [29] and Arithmetic coding [30] are the most popular methods. Both coding methods work on a statistical basis but in a different manner. Huffman coding interprets each symbol

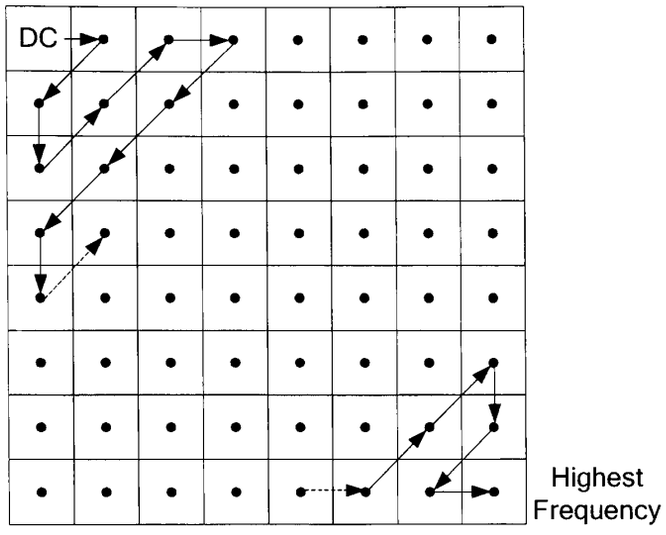


Fig 2-5 DCT coefficient zigzag scan for an 8x8 block (non-interlaced frame).

individually, whereas Arithmetic Coding assigns codeword for a cluster of data [31, 32] such that each symbol can be expressed with fractional bits.

As for the coding of the motion vector, a predicted motion vector (PMV), $\vec{p} = (p_x, p_y)$, is initially computed by taking the median value of the adjacent motion vectors at top, top-left and left. If one of these is unavailable, the motion vector at that position is substituted with zeros. The motion vector difference (MVD) for each macroblock is obtained by subtracting the current motion vector for its PMV. Finally, the MVD is encoded using a variable length code.

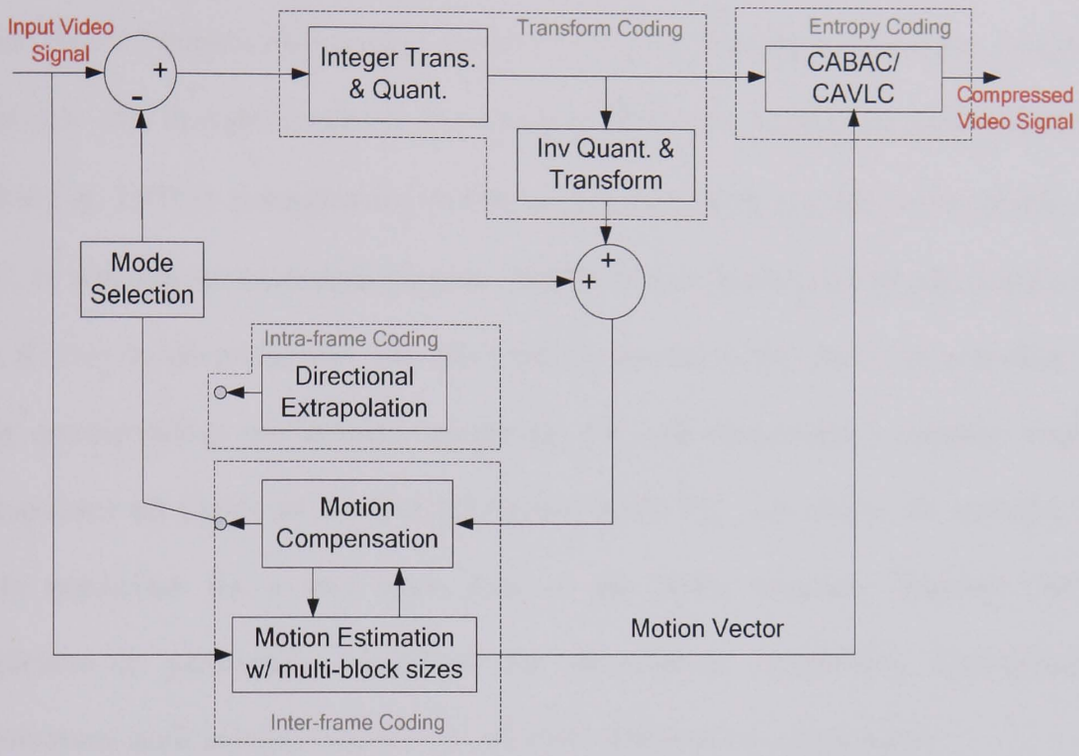
2.3 Structure of H.264/AVC Standard

The hybrid coding structure of the H.264/AVC standard, as shown in Fig. 2-6, is described as a parallel to that of the generic codec in the previous section. The differences are outlined as follows:

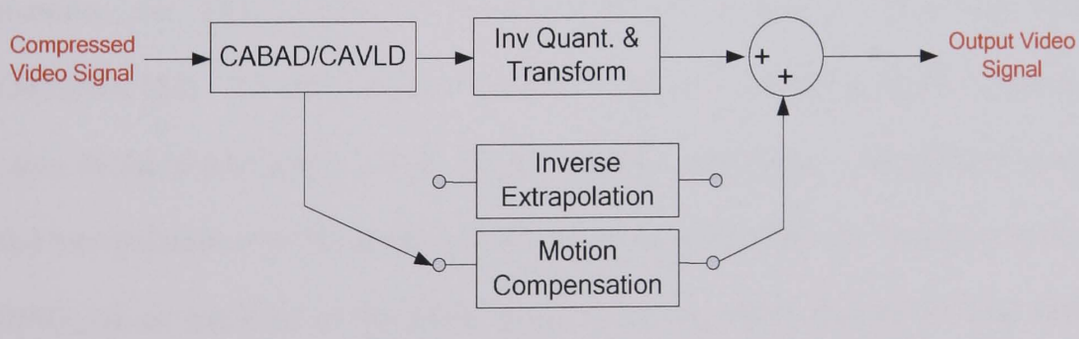
- *Intra-frame coding*: a scheme based on directional extrapolation is employed to predict the spatial composition of each 4x4 block or 16x16 block. *Transform coding* utilising the integer discrete cosine transform (IntDCT) provides spatial-frequency mapping without multiplication operations.
- *Inter-frame coding*: blocks with smaller partition size are incorporated to increase the accuracy of the motion search operation. The variation of multiple block-decomposition provides a set of options/modes. The mode selected is that which produces the least Lagrangian cost.
- *Entropy Coding*: flexible coding schemes, such as Context-based Adaptive Variable Length Coding (CAVLC) and Binary Arithmetic Coding (CABAC), replace conventional Huffman and Arithmetic coding approaches.

2.3.1 Intra-Coding with Directional Extrapolation

In contrast to the DPCM technique, the H.264/AVC standard provides in total 13 intra-predictions to reconstruct the spatial composition of a 4x4 block and a 16x16 block. Block-based spatial prediction results in a better approximation to the original pixel samples due to the increase in the number of neighbouring samples involved in the extrapolation process. The errors that arise from incorrect intra-prediction are further processed with the integer discrete cosine transform (IntDCT) without the need for multiplication computations.



(a)



(b)

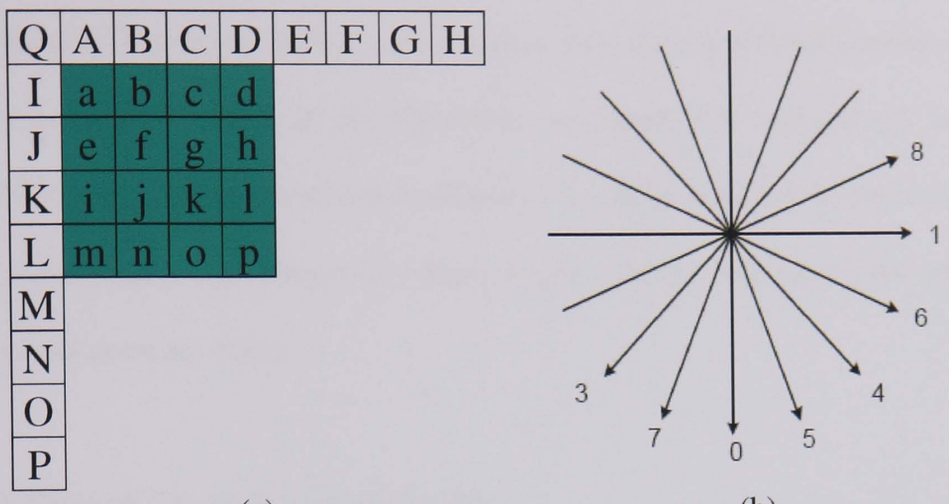
Fig. 2-6 Block structure of the H.264 codec: (a) encoder, (b) decoder.

Intra-frame prediction with Directional Extrapolation

H.264/AVC Intra-frame prediction supports two extrapolation options[1]: *I4MB* and *I16MB*. *I4MB* consists of nine ways to reconstruct the spatial composition of a 4x4 block. Fig. 2-7 depicts elements (a – p) of a 4x4 block with the neighbouring encoded pixels (A – M) in eight directions. For instance, *VERT*, the vertical member (indicated as 0 in Fig. 2-7(b)), extrapolates a 4×4 block vertically with 4 neighbouring pixels, *A*, *B*, *C*, *D*, whereas the horizontal member, *HORT*, utilizes the horizontal adjacent pixels, *I*, *J*, *K*, *L* to do the prediction. The other modes operate in the same way according to their corresponding orientations, except for *DC*, the directionless member, which extrapolates all pixels as $(A+B+C+D+I+J+K+L)/8$. Fig. 2-8 shows an example of *I4MB* prediction for a 4x4 block. Due to the better accuracy obtained, *I4MB* prediction is particularly beneficial for descriptions comprising high-detailed information, such as the boundaries of an object and texture composition.

I16MB resembles *I4MB* but is less time-consuming. Four extrapolation orientations, i.e., *VERT*, *HORT*, *DC*, and *PLANE* are suggested in the Joint Video Specification [33]. The first three schemes are similar to those of *I4MB* except that the area of the prediction is 16x16 pixels. *PLANE* prediction is obtained by linear spatial interpolation with the upper and left-hand samples. Thus, its direction is close *to DIAG_DL* as specified in the *I4MB* option. In terms of the amount of data, *I4MB* requires transmitting 16 times the amount of prediction information compared with *I16MB*. Thus, *I16MB* prediction is more suitable for use in areas of homogeneous content, not needing a detailed description.

[1] In the latest H.264/AVC Fidelity Range Extension (FREXt) [41], *I8MB* coding option is available for intra-frame coding.



(a) (b)

Fig. 2-7 (a) 4x4 block with elements (*a* to *p*) and neighbouring pixels (*A* to *M*); (b) Eight direction-biased intra-modes in a 4x4 block. (0: *VERT*; 1: *HORT*; 2: *DC*, 3: *Diag_DownLeft*; 4: *Diag_DownRight*; 5: *VERT_Right*; 6: *HORT_Down*; 7: *VERT_Left*, 8: *HORT_UP*).

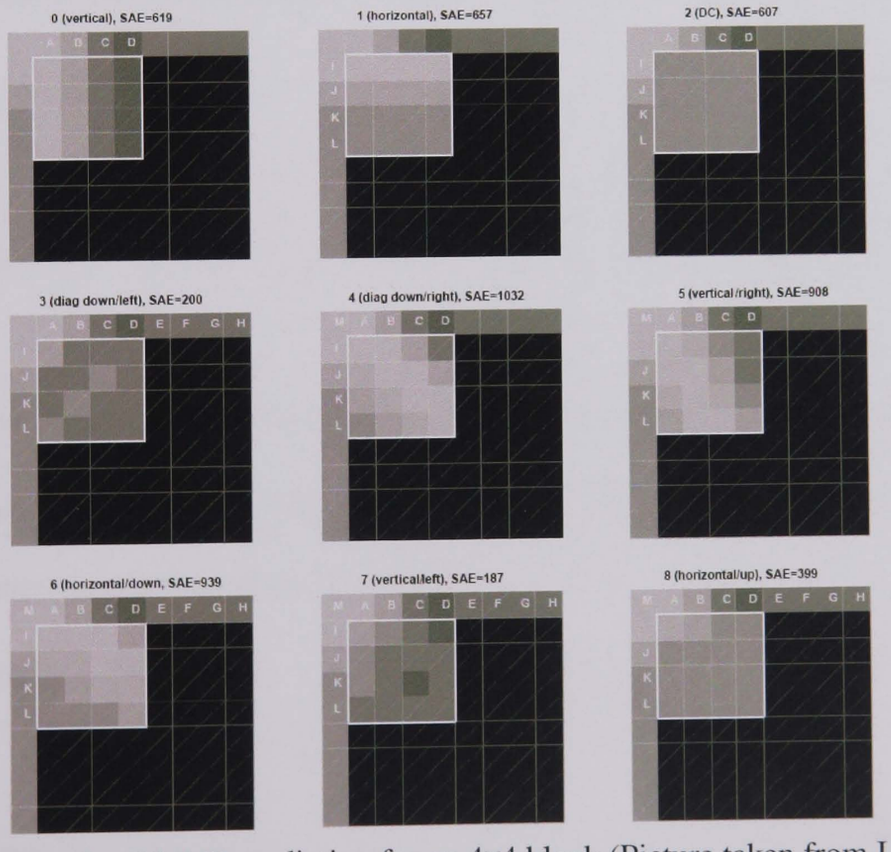


Fig 2-8 Example of *I4MB* prediction for an 4x4 block (Picture taken from I. G. Richardson, H.264 and MPEG-4 Video Compression, Wiley, 2003).

Integer Transform coding

In H.264/AVC transform coding, a 4x4 integer transform is utilised instead of the 8x8 floating point DCT. Since all the operations are carried out with integer arithmetic, loss of accuracy for the transform coefficients is totally avoided. Moreover, an exact inverse operation for the integer transform ensures that the mismatch between encoder and decoder does not occur.

Integer Discrete Cosine Transform (IntDCT)

The IntDCT possesses a pure integer core which approximates to that of the 4x4 DCT [34, 35, 36, 37, 39]. It provides a multiply-free operation without loss of coding gain. According to the definition, the IntDCT is achieved by multiplications with the separable IntDCT matrix cores, $\mathbf{C}_{\text{IntDCT}}$, and its transpose matrix, $\mathbf{C}^{T}_{\text{IntDCT}}$, as

$$\tilde{\mathbf{W}}_f = \mathbf{C}_{\text{IntDCT}} \cdot \mathbf{w}_t \cdot \mathbf{C}^{T}_{\text{IntDCT}}, \tag{2-8}$$

$$\text{where } \mathbf{C}_{\text{IntDCT}} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{pmatrix} \tag{2-9}$$

and $\tilde{\mathbf{W}}_f$ and $\mathbf{w}_t$ denote the matrices representing the transform coefficient and signal block or residue block of size 4x4, respectively. The inverse for the forward IntDCT defined in equation (2-8) as given by [34]:

$$\mathbf{w}_t = \mathbf{C}^{-1}_{\text{IntDCT}} \cdot \tilde{\mathbf{W}}_f \cdot \mathbf{C}^{-T}_{\text{IntDCT}}, \tag{2-10}$$

$$\text{where } \mathbf{C}^{-1}_{\text{IntDCT}} = \begin{pmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{pmatrix} \tag{2-11}$$

Scaling and Quantisation

After the transformation, scaling and quantisation are required to perform lossy coding. Both requirements can be integrated into one equation to reduce unnecessary multiplication and division operations [39, 40]. Let us denote $\tilde{\mathbf{W}}_q$ and $\tilde{\mathbf{W}}_f$ as the quantised and transformed coefficients of the current residue block, respectively. According to the definition,

$$\tilde{W}_q(i,j)=sign\{\tilde{W}_f(i,j)\}\left\lfloor \frac{\tilde{W}_f(i,j).M(Qp\ \%\ 6,r)}{2^{15+Qp/6}}+f \right\rfloor, \qquad (2\text{-}12)$$

where $0\leq i,j\leq 3$ and $r = 2-(i\ \%2)-(j\ \%2)$

The symbols % and $\lfloor \bullet \rfloor$ are the modular and floor operators, respectively; f represents a dead-zone control parameter which is fixed as 1/3 in the intra-coding case and as 1/6 in the inter-coding case [33]; and $M(Qp\%6,\ r)$, the quantisation coefficient for the scaling function, has been pre-defined for each frequency component as a periodic table, M.

$$M = \begin{matrix} r=0 & r=1 & r=2 \end{matrix}\begin{bmatrix} 5243 & 8066 & 13107 \\ 4660 & 7490 & 11916 \\ 4194 & 6554 & 10082 \\ 3647 & 5825 & 9362 \\ 3355 & 5243 & 8192 \\ 2893 & 4559 & 7282 \end{bmatrix}\begin{matrix} \leftarrow (Qp\%6)=0 \\ \leftarrow (Qp\%6)=1 \\ \leftarrow (Qp\%6)=2 \\ \leftarrow (Qp\%6)=3 \\ \leftarrow (Qp\%6)=4 \\ \leftarrow (Qp\%6)=5 \end{matrix}, \qquad (2\text{-}13)$$

The denominator in (2-12) can be expressed as $2<<(15+Qp/6)$, where '$<< N$' symbol indicates an N-bit left shift. The equation can be rewritten in order to remove the division and floor operations:

$$\tilde{W}_q(i,j)=sign\{\tilde{W}_f(i,j)\}.\left[\left(\tilde{W}_f(i,j).M(Qp\ \%\ 6,r)+f.2^{15+Qp/6}\right)>>(15+Qp/6)\right] \qquad (2\text{-}14)$$

and its reconstruction formula becomes:

$$\tilde{W}_f(i,j)=\left[\tilde{W}_q(i,j)\,S(Qp\%6,\,r)\right]<<(Qp/6) \tag{2-15}$$

where $S(Qp\,\%\,6,\,r)=\begin{bmatrix}10 & 16 & 13\\ 11 & 18 & 14\\ 13 & 20 & 16\\ 14 & 23 & 18\\ 16 & 25 & 20\\ 18 & 29 & 23\end{bmatrix}$, $r = 2-(i\,\%2)-(j\,\%2)$ (2-16)

2.3.2 Motion Estimation with Multiple Partitioned Blocks

H.264/AVC inter-frame coding incorporates a motion search and rate-distortion optimisation to provide compression in a temporal fashion. The improvement of motion estimation is achieved with the support of new features, for example, smaller decomposed blocks (of size 4x4, 4x8, 8x4, and 8x8 pixels), multiple reference frames, and motion vectors with quarter-pixel precision. Since each macroblock is processed independently, the encoder also allows the coding of intra-blocks within inter-frames. The choice between intra-coding and inter-coding for the current macroblock is further determined by Lagrangian rate-distortion evaluation.

Motion Estimation Utilising Multiple Partitioned Blocks

Similar to its predecessors, H.264/AVC also relies on the BMA to perform motion estimation. The difference, however, is that the block decomposition is elaborated to macroblock-type and subblock-type. The macroblock-type may be of size 16x16, 16x8, and 8x16 pixels. As for the subblock-type, *P8x8*, comprises a number of smaller partition sizes ranging from 4x4 to 8x8 pixels to constitute a macroblock as a whole. Fig. 2-9 illustrates the partition sizes available for motion estimation. Note that each of

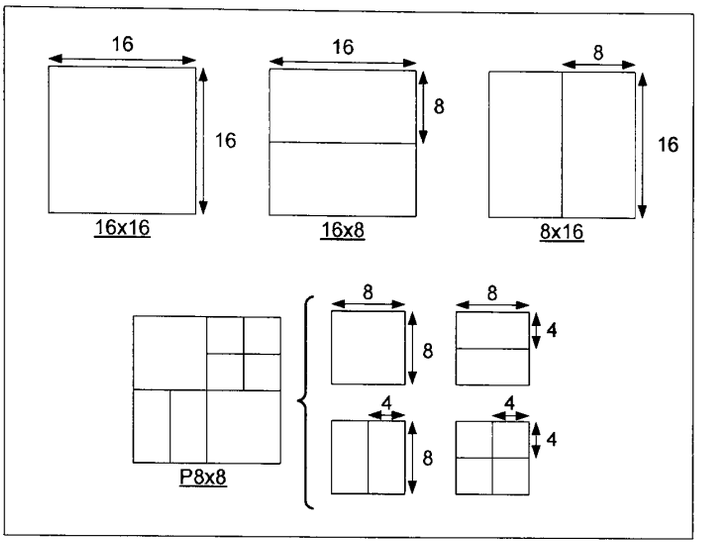


Fig.2-9 Two types of block decompositions for motion estimation: macroblock-type (top) and subblock-type (bottom).

these blocks contains its own motion vector. Consequently, the mode with small partitioned blocks is able to provide a more precise description of disparate motion within a macroblock. Furthermore, the macroblock allows the reconstruction of its decomposed blocks of size 16x8 pixels or 8x16 pixels taken from different reference frames (support up to 5 frames). Finally, the representation of the motion vector is of quarter-pixel precision to benefit the motion estimation in a reduced size picture. Quarter-pixel precision is obtained by a linear interpolation between two half-pixel positions.

Mode Selection and Lagrangian Optimisation

The H.264/AVC encoder allows the co-existence of intra-blocks within an inter-frame. The variation of the decomposition structures, as well as other hybrid coding techniques, provides a set of options/modes for the encoder. The possible prediction modes for an inter-block are

$$\text{mode} \in \{SKIP, I4MB, I16MB, 16x16, 16x8, 8x16, P8x8\} \qquad (2\text{-}17)$$

where *I4MB* and *I16MB* are the intra-prediction options described in Section 2.3.1, *SKIP* is the mode that directly copies the content of the macroblock at the same position in the reference frame (denoted as the co-located macro-block), without the need for motion compensation. In the H.264/AVC standard, an evaluation incorporating Lagrangian rate-distortion optimisation is employed to select the best prediction mode from (2-17). Let us denote $\boldsymbol{w}_t$ as a macroblock in a frame at time t and $\hat{\boldsymbol{w}}$ as a reconstructed block of $\boldsymbol{w}_t$. Then, the macroblock-based Lagrangian cost, J_{MB}, [17, 41] is

$$J_{\mathrm{MB}}(\boldsymbol{w}_t, \hat{\boldsymbol{w}}, \mathrm{mode} \mid Qp) = D(\boldsymbol{w}_t, \hat{\boldsymbol{w}} \mid Qp) + \lambda_{\mathrm{MODE}} . R(\boldsymbol{w}_t, \hat{\boldsymbol{w}} \mid Qp) \quad (2\text{-}18)$$

where λ_{MODE} is a Lagrangian multiplier associated with the quantisation factor, Qp, by the relationship

$$\lambda_{\mathrm{MODE}} = 0.85 \times 2^{Qp/3} \quad (2\text{-}19)$$

In (2-18), R reflects the number of bits, including the macroblock header, motion vectors, and all residue data, associated with the chosen mode and Qp. D denotes a distortion measure quantifying the difference between $\boldsymbol{w}_t$ and $\hat{\boldsymbol{w}}$, restored by motion compensation,

$$D(\boldsymbol{w}_t, \hat{\boldsymbol{w}} \mid Qp) = \sum_x \sum_y \left(w_t(x, y) - \hat{w}(x, y)\right)^2 \quad (2\text{-}20)$$

A simplified Lagrangian evaluation has also been developed for the subblock selection within the *P8x8* mode. The equation is denoted as

$$J_{sub}(\boldsymbol{w}_t, \hat{\boldsymbol{p}}_{t-\tau}) = \mathrm{SAD}(\boldsymbol{w}_t, \hat{\boldsymbol{p}}_{t-\tau}) + \lambda_{\mathrm{MOTION}} . R(\vec{m} - \vec{p}) \quad (2\text{-}21)$$

$$\lambda_{\mathrm{MOTION}} = \sqrt{\lambda_{\mathrm{MODE}}} \quad (2\text{-}22)$$

where $\hat{\boldsymbol{p}}_{t-\tau}$ is a predicted subblock taken from frame t-τ restored by the motion vector, $\vec{m}$; $(\vec{m} - \vec{p})$ is the MVD for the current block (see Section 2.2.3). In (2-21), one notices that the computation of the distortion component has become the sum of absolute

difference between the original subblock and the predicted subblock, while the rate parameter is replaced by the bits spent to encode the MVD.

The final decision for both (2-18) and (2-21) is determined by the mode that produces the least Lagrangian cost. Currently, the H.264/AVC standard employs a brute force algorithm to find the optimum mode and its motion vector(s) to achieve the best compromise between distortion and bit rate.

2.3.2 Context-based Adaptive Coding

The H.264/AVC standard supports two advanced entropy coding methods, CAVLC (context-adaptive variable length coding) [38] and CABAC (context-based adaptive binary arithmetic coding) [43], which can adaptively adjust the performance according to the context of the source materials.

The CAVLC algorithm follows a similar principle to that of Huffman coding, i.e., the length of codeword is inversely proportional to the occurrence frequency of the symbol. The difference is that Huffman coding treats each non-zero coefficients as an individual symbol; while the CAVLC considers a transformed block as a whole. This block-orientation allows CAVLC to treat some combinations of symbols (e.g., a sequence of ±1) as special cases, with fewer bits used to represent them.

As for the CABAC technique, it contains a complicated structure incorporating an arithmetic coding engine as illustrated in Fig. 2-10. Binarisation is a conversion process converting integer symbols into a pre-defined binary representation called a bin string. For each bin string, a context index is defined and assigned according to the occurrence of the bins. Finally, the bin string is encoded by an arithmetic coding engine using corresponding probability tables addressed by the

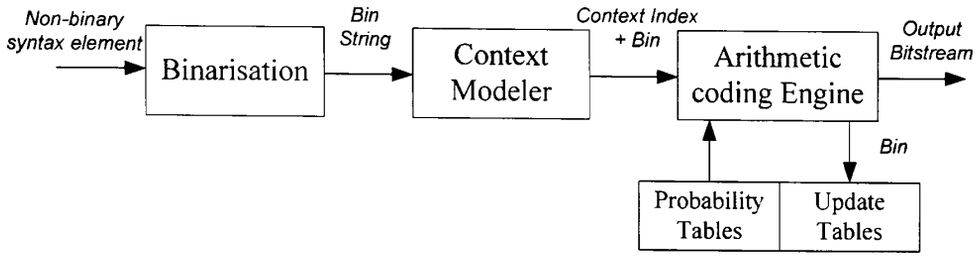


Fig. 2-10 Simplified block diagram of a Context-based Adaptive Binary Arithmetic Coding (CABAC) encoder.

context index. The probability tables are updated accordingly by the bins currently being processed.

In terms of performance, the utilisation of CABAC ensures a 10% to 15% bit-rate saving compared to that of the CAVLC, but it is achieved at the cost of a complex implementation [51].

2.4 Performance Comparison

This section reviews the performance of the H.264/AVC coding technique and other algorithms, such as MPEG-2, H.263+, and MPEG-4. Prior to this, the test material and objective quality assessment method used are introduced.

2.4.1 Test Material

The test material employed for the experimental evaluation are of two resolutions, i.e., CIF (Common Intermediate Format, 352x288 pixels) and QCIF (Quarter Common Intermediate Format, 176x144 pixels). Each sequence is categorised according to one of three classes recommended by the MPEG-4 Verification Model (VM) [47] depending on the spatial detail and motion:

A – Low spatial complexity and little motion,

B – Medium spatial complexity and low amount of motion, or vice versa.

C – High spatial complexity and medium amount of motion activity, or vice versa.

Appendix 1 lists all the test material used in this thesis, according to the class of sequence. Each frame of the test sequences contains one luminance component (denoted as Y) and two colour/chrominance components (Cr and Cb, or popularly known as U and V) [49]. The conversion formulae between YCrCb and RGB are:

$$\begin{aligned} Y &= 0.257R+0.504G+0.098B+16 \\ C_b &= -0.148R-0.291G+0.439B+128 \\ C_r &= 0.439R-0.368G-0.071B+128 \end{aligned} \tag{2-23}$$

In contrast to the RGB representation, the YUV coordinate system has preferential energy concentrated in the luminance component [48]. To avoid over-representation of the colour information, the spatial distribution of Y, U, and V components are

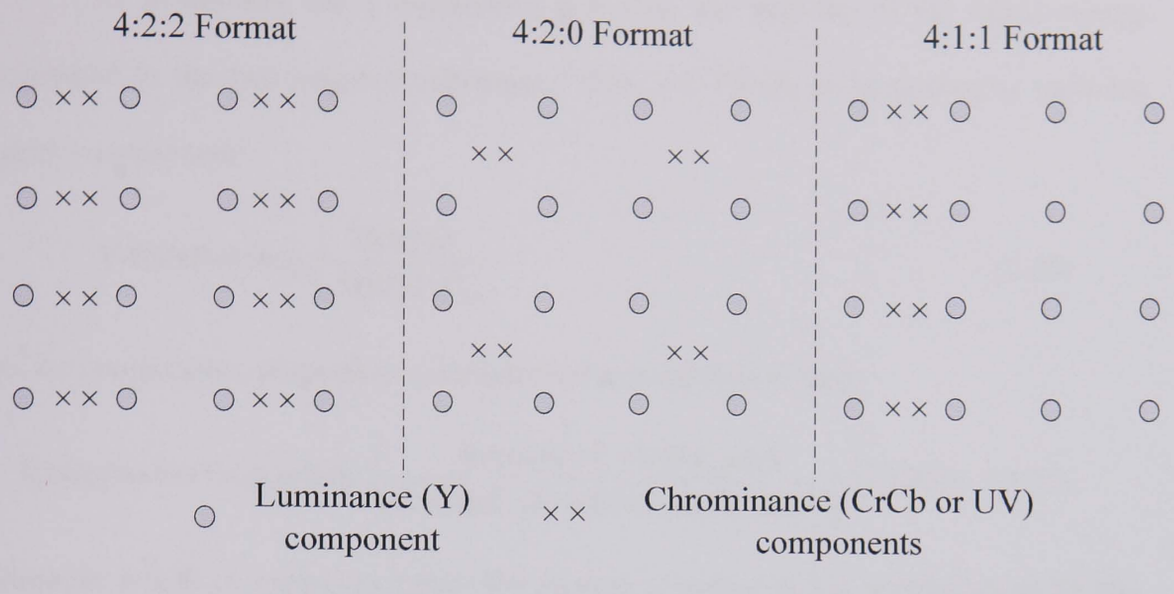


Fig. 2-11 Spatial distribution of luminance (Y) and chrominance (CrCb) components to form different formats for a colour picture.

normally arranged in different formats as illustrated in Fig. 2-11. Except where stated, all the test sequences employed for the experimental evaluation are in 4:2:0 format.

2.4.2 Objective Quality Assessments

Two objective metrics have been developed to evaluate the quality of coded pictures and the compression ratio between output data and source material. In terms of picture quality assessment, Peak Signal-to-Noise Ratio (PSNR, measured in dB), is a popular choice. Calculation of PSNR value requires the MSE (mean square error) defined in the (2-6):

$$\mathrm{PSNR}=10\log_{10}\left(\frac{255\times255}{\mathrm{MSE}(Y,\hat{Y},\ U,\ \hat{U},\ V,\ \hat{V})}\right) \tag{2-24}$$

$$\mathrm{MSE}(Y,\hat{Y},\ U,\ \hat{U},\ V,\ \hat{V})=\frac{4\times\mathrm{MSE}(Y,\hat{Y})+\mathrm{MSE}(U,\hat{U})+\mathrm{MSE}(V,\ \hat{V})}{6} \tag{2-25}$$

where Y, U, and V represent the intensity of the original image Y, U, and V components, respectively; whereas $\hat{Y}$, $\hat{U}$, and $\hat{V}$ are the corresponding component reconstructions.

As mentioned, the Y component possesses the majority of the signal energy compared to the two colour components. Thus, the PSNR of luminance is used for quality assessment:

$$\text{Y-PSNR}=10\log_{10}\left(\frac{255\times255}{\text{MSE}(Y,\hat{Y})}\right) \tag{2-26}$$

As for compression proportion, a straightforward equation is used:

$$\text{Compression Proportion}=\left(\frac{\text{Amount of output data}}{\text{Amount of source material data}}\right)\times100\% \tag{2-27}$$

Since the length of sequences varies, the amount of output data is normalised as bit rate to identify the average bits spent for each second (measured in Kbits/sec). Note that the value of bit rate inversely reflects the compression ratio.

$$\text{Bit Rate}=\frac{\text{Total bits of output data}}{\text{Playing length of the source material}} \tag{2-28}$$

Finally, it is worth mentioning that several fast algorithms for the H.264/AVC coding standard are developed in Chapters 3, 4, and 5. In order to calculate the achieved speedup, we denote $T_{\text{H.264}}$ and T_{fast} as the total computational time required by the H.264/AVC encoder and each fast algorithm, respectively. The reduction in coding time, ΔT, is expressed as a percentage

$$\Delta\text{T}=\left(1-\frac{T_{fast}}{T_{\text{H.264}}}\right)\times100\% \tag{2-29}$$

2.4.3 Performance Comparisons

Fig. 2-12 shows the performance of four different video coding techniques, H.264/AVC, MPEG-4, MPEG-2, and H.263 using the *Foreman* (top) and *Tempete* (bottom) sequences. The results are presented as rate-distortion diagrams where bit

Table 2-1
Average bit-rate savings of H.264/AVC in comparison with other standards.

Coders	MPEG-4	H.263	MPEG-2
H.264/AVC	38.62%	48.80%	64.46%
MPEG-4	--	16.65%	42.95%
H.263	--	--	30.61%

* Original data obtained from [Schäfer03]

rate and PSNR of luminance (Y-PSNR) are the units for the X and Y coordinates, respectively. As mentioned, PSNR is a metric reflecting a better perceptual quality at higher values, whereas low bit rate represents fewer bits spent for each second of play. These two quality assessments are in principle a trade-off. In the rate-distortion diagram, the coding performance of each coding method is indicated by the gradient of its corresponding curve, i.e., a greater gradient indicates a better coding efficiency.

From Fig. 2-12, it is clear that the red curves, representing H.264/AVC, achieve an exponential increase with the greatest gradient when compared to the other methods. It demonstrates that the coding efficiency of H.264/AVC outperforms other methods in terms of picture quality and bit rate. Table 2-1 shows the average bit rate savings provided by four video coding standards, with respect to other tested encoders over the entire set of sequences and bit rates. One can see that, the H.264/AVC algorithm achieves bit reductions of 38.62%, 48.80%, and 64.46%, compared to MPEG-4, H.263, and MPEG-2.

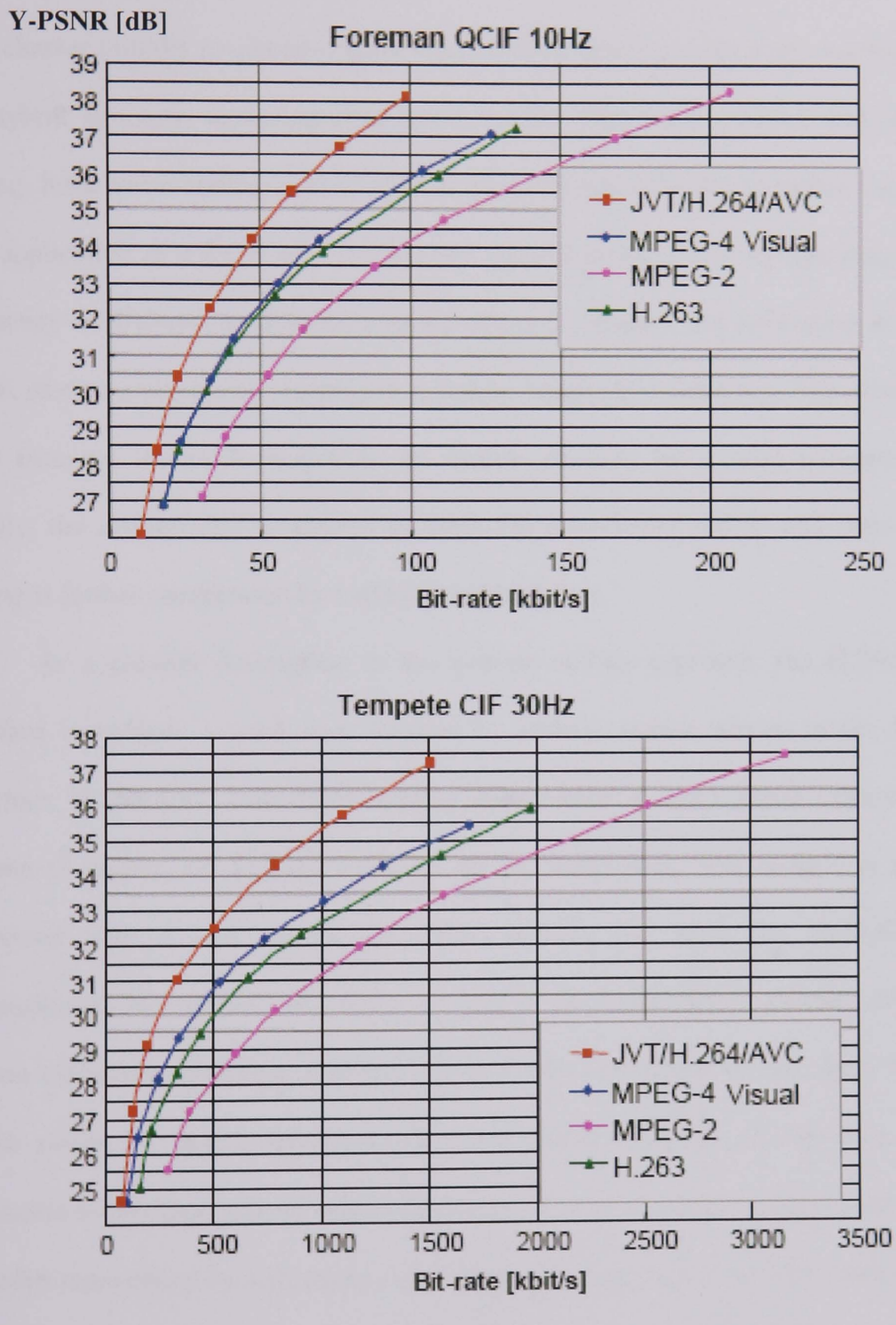


Fig. 2-12 Performance of four video coding standards, H.264/AVC, MPEG-4, H.263, and MPEG-2 for the *Foreman* (top) and *Tempete* (bottom) sequences (diagram taken from [44]).

2.5 Summary

This chapter initially reviewed generic video coding, placing particularly emphasis on the hybrid structure, including intra-frame coding, inter-frame coding, and entropy coding. Intra-frame coding aims to remove the pixel-based similarity within a picture. The application of a DCT and quantisation achieve compression by allowing fewer frequency coefficients to describe a group of pixel elements. As for coding an inter-block, motion estimation is employed to reduce temporal redundancy. The errors that arise from an incorrect prediction are further encoded by motion compensation. Finally, the residue data generated by both the intra-frame coding and inter-frame coding is further compressed by variable length coding.

As a parallel description to the generic coding approach, the H.264/AVC standard introduces several new features to address improvements in the hybrid structure. H.264/AVC intra-frame coding incorporates a block-based extrapolation scheme to reduce the average intensity for an intra-block. The residue is further processed with a 4x4 integer transform without the need for multiplication computations. In contrast, inter-frame coding in the H.264/AVC structure employs motion estimation featuring multiple partition blocks ranging in size from 4x4 to 16x16 pixels. A search incorporating small partition blocks results in a better adaptation to the local activity within the image. The mode decision for a macroblock is further determined by Lagrangian evaluation which maintains the best compromise between rate and distortion. At the end of the section, two context-based coding schemes, CABAC and CAVLC, were briefly introduced.

In Section 2.4, the performance of the H.264/AVC codec is summarised in comparison with other video coding methods. Comparisons are given for Y-PSNR (measured in dB) and bit rate (Kbits/sec). The results demonstrate that the

H.264/AVC algorithm has significantly improved the rate-distortion performance of that achieved by other video coding standards. Moreover, a bit reduction of up to 64% is obtained by the H.264/AVC algorithm when encoding pictures of similar quality.

Chapter 3

Frequency Domain Approach To Fast Intra-frame Coding

The H.264/AVC hybrid structure has demonstrated improved performance over existing video coding standards. However, this is achieved at the cost of additional complexity for both encoder and decoder architectures [51, 52, 53, 54]. Schäfer et al. [51] identified that the computational burden imposed by the H.264/AVC encoder is approximately 4-5 times more than that of MPEG-2 [51, 55]. In practice, the design of real-time H.264/AVC applications may be amenable to hardware solutions. However, the inflexibility of a VLSI chip design introduces restrictions for the development of efficient algorithms. Consequently, on-going developments in software-based H.264/AVC codecs seek to achieve a computational reduction without significant degradations in rate and distortion.

In the past, research effort has mainly contributed to the Joint Model (JM) reference software, a laboratory implementation of the H.264/AVC standard. More precisely, fast coding algorithms for both intra-frames and inter-frames have become the main topic in this thesis. In this chapter, a frequency domain approach to achieve efficient intra-frame coding is detailed. The next two chapters examine several algorithms for multiple inter-mode selection.

The remainder of this chapter is organised as follows: A literature review on fast intra-frame coding is discussed in the next section. Section 3.3 gives a detailed formulation of the proposed fast intra-frame mode selection, *Fintra*, algorithm. The results of simulations are summarised in Section 3.4. Finally, Section 3.5 discusses some overall conclusions.

3.1 Literature Review

In the H.264/AVC encoder, an intra-mode is selected that produces the least Lagrangian cost. The computation of Lagrangian cost for each mode necessitates valuable computational resources due to the requirements of image reconstruction and bits-spent estimation, as indicated in (2-18). Thus, a number of suggestions have been made to reduce processing time for intra-frame coding. The most straightforward way is to develop a pre-selection process to shortlist fewer candidates for Lagrangian examination. Generally, most of the pre-selection algorithms are applied in the spatial (pixel) domain. Few studies have been done in other areas, for example, prediction in the temporal domain [68] and utilising a joint spatial-frequency decision [66, 67].

Existing solutions for intra-frame coding are summarised as follows. Meng et al. [57, 58] selected an intra-mode by computing a partial cost for down-sampled pixels instead of for a 4x4 block. However, this algorithm lacked integration with the Lagrangian evaluation. Chen, Lin and Chang [59, 60] developed a step-search strategy for *I4MB* mode selection. In each step, the algorithm considers only two nearest extrapolation directions corresponding to the outcome acquired from the last search. This strategy results in good prediction, yet the speedup obtained is less satisfactory. Pan et al. [61] proposed a fast algorithm based on edge information [63]

and a directional field [62]. A recently-published improvement in [64] obtained a good reduction in computational requirement.

As for alternative approaches, the algorithm proposed by Xin and Vetro [68] makes use of the temporal correlation between successive frames. By reusing the mode decision obtained from the collocated block in the previous frame, the mode is determined without a need for further examination. This temporal domain approach does not produce desirable results in both rate-distortion performance and speedup requirement. However, it opens a new dimension in fast intra-frame coding. In contrast, Kim et al. [66, 67] advocated an algorithm operating in the Hadamard transform domain. The final intra-mode is selected by a joint spatial and frequency decision. The result demonstrated a good coding efficiency. However, the picture distortion and bits overhead arising from incorrect predictions are noticed at high bit rates.

3.2 Proposed Frequency Domain Approach

A fast intra-mode selection (*Fintra*) algorithm has been developed. The success of the proposed algorithm is achieved by selecting fewer modes to undergo full Lagrangian examination. Generally, the residues of intra blocks have relatively large block energy due to the spatial prediction from neighbouring pixels. Consequently, it is more efficient to operate in the frequency domain rather than in the spatial domain. Furthermore, it is observed that the modes that provide the least residue energy will also result in minimum rate R and hence minimise the Lagrangian cost. The chart in Fig. 3-1 illustrates this observation. The chart shows the relationship between the candidates selected by distortion cost and those by Lagrangian evaluation. Results were obtained by coding thirty frames of three test sequences in CIF format (352×288 pixels), *Akiyo*, *Foreman*, and *Mobile & Calendar*, representing three different degrees of spatial correlation and movement. The *I4MB* intra-coding option was used. The most probable modes (MPM), the candidates predicted from prior knowledge of

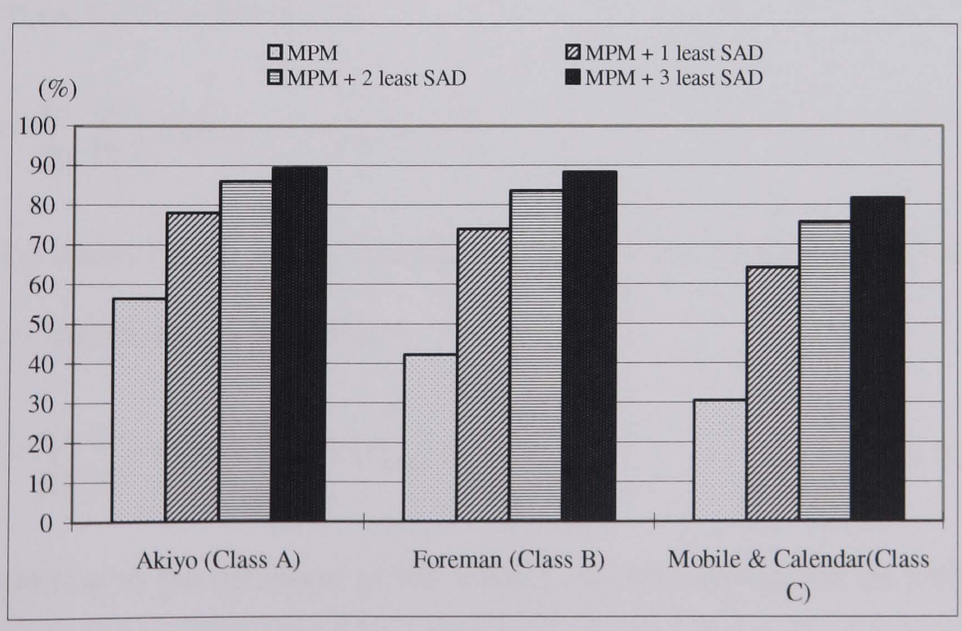


Fig. 3-1 Match percentage between the least distortion cost acquired from SAD implementation and the least rate-distortion cost obtained from Lagrangian evaluation.

neighbouring blocks, account for 56%, 42% and 30% of the mode decisions made by Lagrangian evaluation, for the three test sequences respectively. These percentages increase to 89%, 88% and 81% respectively when three more candidates, each selected as having the lowest residue energy, are chosen.

To reduce the computational cost of expensive Lagrangian cost evaluation, we can limit the number of members (say M) that need to undergo the full evaluation process. The M members are those with the least residue energy from amongst all the possible members. The entire selection process operates in the integer discrete cosine transform (IntDCT) domain.

3.2.1 Algorithm Formulation

The selection criterion is the least residue energy which can be measured from the SAD outcome between the original block and predicted block. First, let us denote a 4×4-pixels original block as $\boldsymbol{B}$ and any intra predicted block as $\boldsymbol{P}_{\text{mode}}$. The error cost for the residue block is measured as

$$\varepsilon = \sum_{m=0}^{3} \sum_{n=0}^{3} \left| \mathcal{T}\{\boldsymbol{B}(m,n) - \boldsymbol{P}_{\text{mode}}(m,n)\} \right| \tag{3-1}$$

where $\mathcal{T}\{.\}$ stands for the integer discrete cosine transform (IntDCT). Since IntDCT is a unitary transformation, we obtain

$$\varepsilon = \sum_{m=0}^{3} \sum_{n=0}^{3} \left| \mathcal{T}\{\boldsymbol{B}(m,n)\} - \mathcal{T}\{\boldsymbol{P}_{\text{mode}}(m,n)\} \right| \tag{3-2}$$

Then, according to the definition of the IntDCT, we can decompose the frequency spectra of each block as:

$$\varepsilon = \left|DC_{\mathrm{B}} - DC_{\mathrm{mode}}\right| + \left|AC_{B}(1,0) - AC_{\mathrm{mode}}(1,0)\right| + \left|AC_{B}(0,1) - AC_{\mathrm{mode}}(0,1)\right| + \ldots + \left|AC_{B}(3,3) - AC_{\mathrm{mode}}(3,3)\right| \tag{3-3}$$

where DC represents the low frequency coefficient of a DCT block and $AC(m,n)$ accounts for the high frequency coefficient located at the (m,n) position of the frequency block, where $0 \le m, n \le 3$ except $m = n = 0$. Since the optimal mode is selected with the least error cost, $\varepsilon_{\min}$, we obtain

$$\text{optimal mode} = \underset{\text{mode}}{arg\,min}\left\{\left|DC_{B} - DC_{\mathrm{mode}}\right| + \sum_{\substack{0\le m,n\le 3 \\ (m,n)\neq 0}}^{\mathrm{num}(AC)=15}\left|AC_{B}(m,n) - AC_{\mathrm{mode}}(m,n)\right|\right\} \tag{3-4}$$

where num(AC) accounts for the number of AC coefficients.

Equation (3-3) further indicates that an approximation of ε can be made provided that enough AC coefficients are selected. It is observed that the low-frequency AC coefficients normally contain more energy than high-frequency coefficients. Consequently, we select only a few of the first low-frequency AC coefficients from the DCT block, let's say, $AC(1,0)$, $AC(0,1)$, $AC(1,1)$, $AC(2,0)$, and $AC(0,2)$. Then, (3-4) becomes

$$\text{optimal mode} = \underset{\text{mode}}{arg\,min}\left\{\left|DC_{B} - DC_{\mathrm{mode}}\right| + \sum_{\substack{m,n \\ (m,n)\neq 0}}^{\mathrm{num}(AC)<15}\left|AC_{B}(m,n) - AC_{\mathrm{mode}}(m,n)\right|\right\} \tag{3-5}$$

Note that the accuracy of the approximation increases if more coefficients are involved.

By simple calculations from the 2D-IntDCT definition in (2-8), we can easily obtain the formulae of the following DC and $AC(m,n)$ components of a 4×4 block:

$$DC = \sum_{m=0}^{3}\sum_{n=0}^{3} \boldsymbol{B}(m,n) \tag{3-6}$$

$$AC(0,1) = \sum_{m=0}^{3}\left(\boldsymbol{B}(m,1) - \boldsymbol{B}(m,2)\right) + \left\{\sum_{m=0}^{3}\left(\boldsymbol{B}(m,0) - \boldsymbol{B}(m,3)\right)\right\} << 1 \tag{3-7}$$

$$AC(0,1) = \sum_{n=0}^{3}\left(\boldsymbol{B}(1,n) - \boldsymbol{B}(2,n)\right) + \left\{\sum_{n=0}^{3}\left(\boldsymbol{B}(0,n) - \boldsymbol{B}(3,n)\right)\right\} << 1 \tag{3-8}$$

$\vdots$ $\vdots$

where << 1 represents a 1-bit left shift, equivalent to a multiplication by 2.

Next, we consider how to efficiently access the value of the DC_{mode} and the other five $AC_{\text{mode}}(m,n)$ components for each predicted block. Unlike the original block, the predicted blocks are the direction-biased extrapolations from the neighbouring pixels. In order to formulate an efficient calculation, we represent any of (3-6) to (3-8) in matrix form with a' to p', the elements of any predicted block, as

$$AC_{\text{mode}}(m,n) = \tau_{1D}^{(m,n)} \bullet \begin{bmatrix} a' \\ \vdots \\ \vdots \\ p' \end{bmatrix}_{1\times 16} \tag{3-9}$$

where $\tau_{1D}^{(m,n)}$ is an 1D-IntDCT conversion vector between an $AC_{\text{mode}}(m,n)$ and the predicted elements (a' to p'). For instance, $\tau_{1D}^{(0,0)}$ has all the coefficients equal to 1.

In a similar manner, a matrix formula can be provided to relate the predicted elements and the vector, $\vec{\mu}$, which consists of the neighbouring samples (A to Q) as the coefficients.

$$\begin{bmatrix} a' \\ \vdots \\ \vdots \\ p' \end{bmatrix} = \tilde{C}_{\text{mode}} \bullet \vec{\mu} = \begin{bmatrix} C_{1,1} & C_{1,2} & \cdots & C_{1,16} & C_{1,17} \\ C_{2,1} & \ddots & & & C_{2,17} \\ \vdots & & \ddots & & \vdots \\ C_{15,1} & & & \ddots & C_{15,17} \\ C_{16,1} & C_{16,2} & \cdots & C_{16,16} & C_{16,17} \end{bmatrix}_{16\times 13} \bullet \begin{bmatrix} A \\ \vdots \\ \vdots \\ M \end{bmatrix}_{13\times 1} \tag{3-10}$$

where $\tilde{C}_{\text{mode}}$ is a 16-by-17 conversion matrix, for instance, $\tilde{C}_{\text{HORT}}$, the conversion matrix of the horizontal member, extrapolates the horizontal pixel I to the first row's elements, i.e., a' to d'. Then, all the coefficients in the first 4 rows of $\tilde{C}_{\text{HORT}}$ are zero except for the ninth coefficients ($C_{1,9}$, $C_{2,9}$, $C_{3,9}$, $C_{4,9}$, i.e., position of I), which are one.

We then obtain the relationship between the $AC_{\text{mode}}(m,n)$ and the neighbouring pixels (A to Q) by combining (3-9) and (3-10).

$$AC_{\text{mode}}(m,n) = \tau_{1\text{D}}^{(m,n)} \bullet \tilde{C}_{\text{mode}} \bullet \vec{\mu} = \boldsymbol{\omega}_{\text{mode}}^{(m,n)} \bullet \begin{bmatrix} A \\ \vdots \\ \vdots \\ M \end{bmatrix}_{13\times 1} \tag{3-11}$$

where $\boldsymbol{\omega}_{\text{mode}}^{(m,n)}$ is a 1-by-13 transpose vector. By arranging the elements of $\boldsymbol{\omega}_{\text{mode}}^{(m,n)}$ to form a matrix, we can obtain the values of $AC_{\text{mode}}(m,n)$ for all the nine *I4MB* modes.

$$\begin{bmatrix} AC_{HORT}(m,n) \\ AC_{VERT}(m,n) \\ \vdots \\ AC_{HORT\_U}(m,n) \end{bmatrix}_{9\times 1} = \tilde{\boldsymbol{\Omega}}'_{(m,n)} \bullet \begin{bmatrix} A \\ \vdots \\ \vdots \\ M \end{bmatrix}_{13\times 1} \tag{3-12}$$

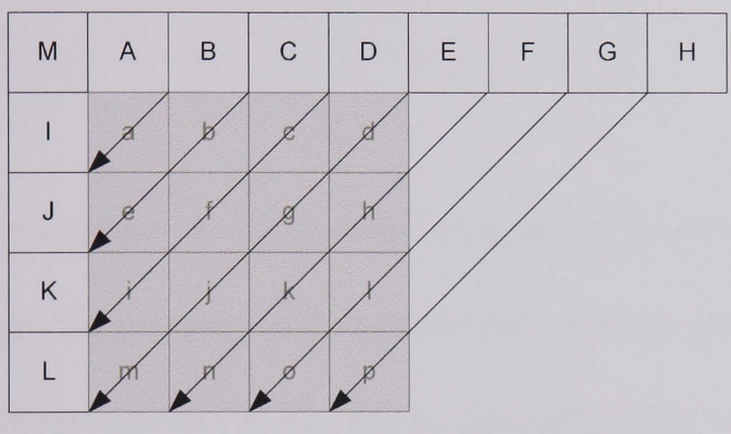
$$\text{Where } \tilde{\boldsymbol{\Omega}}'_{(m,n)} = \begin{bmatrix} \boldsymbol{\omega}_{HORT}^{(m,n)} \\ \boldsymbol{\omega}_{VERT}^{(m,n)} \\ \vdots \\ \boldsymbol{\omega}_{HORT\_U}^{(m,n)} \end{bmatrix}_{9\times 13} \tag{3-13}$$

$\tilde{\boldsymbol{\Omega}}'_{(m,n)}$ in (3-13) is a 9-by-13 sparse matrix. Note that $\tilde{\boldsymbol{\Omega}}'_{(m,n)}$ for all positions (m,n) can be calculated and stored in advance. We integrate the utilisation of $\tilde{\boldsymbol{\Omega}}'_{(m,n)}$ into (3-5), and the selection for the optimal intra-mode becomes

$$\text{optimal mode} = \arg\min_{\text{mode}} \left\{ \left| DC_{\text{B}}.\vec{\mathbf{I}} - \tilde{\boldsymbol{\Omega}}'_{(0,0)}.\vec{\mu} \right| + \sum_{\text{num}(AC)<15} \left| AC_{\text{B}}(m,n).\vec{\mathbf{I}} - \tilde{\boldsymbol{\Omega}}'_{(m,n)}.\vec{\mu} \right| \right\} \tag{3-14}$$

where $\vec{\mathbf{I}}$ is an 9x1 identity vector with unity coefficients. Note that (3-14) arranges the modes with respect to the row position. Thus the optimal mode is selected by the row which possesses the least residue energy.

Example: Evaluating the matrix coefficients of $\boldsymbol{\omega}_{DIAG\_DL}^{(0,0)}$, the DC component of mode 3 (diagonal down-left extrapolation) in *I4MB* is as shown in Fig. 3-2.



$$a = \frac{A + 2B + C + 2}{4}$$
$$e = b = \frac{B + 2C + D + 2}{4}$$
$$c = f = i = \frac{C + 2D + E + 2}{4}$$
$$d = g = j = m = \frac{D + 2E + F + 2}{4}$$
$$h = k = n = \frac{E + 2F + G + 2}{4}$$
$$l = o = \frac{F + 2G + H + 2}{4}$$
$$p = \frac{G + 3H + 2}{4}$$

Fig. 3-2 The extrapolation scheme with diagonal down-left direction and its extrapolation equations for a 4x4 block.

Then, the conversion matrix for DIAG_DL direction, $\tilde{C}_{DIAG\_DL}$, should be

$$C_{DIAG\_DL} = \begin{array}{c} \begin{array}{cccccccccccccc} & A & B & C & D & E & F & G & H & I & J & K & L & M & \\ \end{array} \\ \begin{pmatrix} 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a \\ 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b \\ 0 & 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & c \\ 0 & 0 & 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & d \\ 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e \\ 0 & 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & f \\ 0 & 0 & 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & g \\ 0 & 0 & 0 & 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & h \\ 0 & 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & i \\ 0 & 0 & 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & j \\ 0 & 0 & 0 & 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & k \\ 0 & 0 & 0 & 0 & 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & l \\ 0 & 0 & 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & m \\ 0 & 0 & 0 & 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & n \\ 0 & 0 & 0 & 0 & 0 & 1/4 & 1/2 & 1/4 & 0 & 0 & 0 & 0 & 0 & o \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/4 & 3/4 & 0 & 0 & 0 & 0 & 0 & p \end{pmatrix} \end{array}$$

The $\boldsymbol{\omega}_{DIAG\_DL}^{(0,0)}$ is obtained by performing (3-9):

$$\begin{bmatrix} 1/4 & 1 & 2 & 3 & 7/2 & 3 & 2 & 5/4 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

It is noted that some of the $\boldsymbol{\omega}_{DIAG\_DL}^{(0,0)}$ coefficients possess fractional numbers. To achieve non floating-point computation, we multiply all the coefficients by 4 and obtain

$$\begin{bmatrix} 1 & 4 & 8 & 12 & 14 & 12 & 8 & 5 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The coefficients of $\boldsymbol{\omega}_{\text{mode}}^{(0,0)}$ for other extrapolation directions are acquired in a similar manner. By arranging the elements of $\boldsymbol{\omega}_{\text{mode}}^{(0,0)}$ to form a matrix, we can obtain $\tilde{\boldsymbol{\Omega}}_{(m,n)}$ in (B-1) (see Appendix 2). Note that the relationship between $\tilde{\boldsymbol{\Omega}}'_{(m,n)}$ and $\tilde{\boldsymbol{\Omega}}_{(m,n)}$ is

$$\tilde{\boldsymbol{\Omega}}'_{(m,n)} = (\tilde{\boldsymbol{\Omega}}_{(m,n)} + 2) << 2. \tag{3-15}$$

Thus, we rewrite (3-14) as

$$\text{optimal mode} = \arg\min_{\text{mode}} \left\{ \left| DC_{\mathbf{B}}.\vec{\mathbf{I}} - \left(\tilde{\boldsymbol{\Omega}}_{(0,0)}.\vec{\mu}\right) >> 2 \right| + \sum_{\text{num}(AC)<15} \left| AC_{\mathbf{B}}(m,n).\vec{\mathbf{I}} - \left(\tilde{\boldsymbol{\Omega}}_{(m,n)}.\vec{\mu}\right) >> 2 \right| \right\} \tag{3-16}$$

where >> 2 represents a 2-bit right shift, equivalent to a non floating-point division by 4.

3.2.2 Algorithm description

The proposed algorithm utilizes (3-16) to shortlist *M* candidates from the 9 modes. However, since the experimental investigations detailed in Fig. 3-1 indicate that the most probable mode (MPM) has a higher chance of being selected as an optimal mode, it is included in the short-listed candidates although it may not produce the least residue energy. The proposed algorithm is summarized as follows:

A1. Evaluate (3-6)-(3-8) to obtain the DC and several AC coefficients for the original processed block.

A2. Calculate values of DC_{mode} *and the same number of* $AC_{\text{mode}}(m,n)$ *coefficients for the 9 predicted blocks, by utilizing* $\tilde{\mathbf{\Omega}}_{(0,0)}$ *and* $\tilde{\mathbf{\Omega}}_{(m,n)}$ *in the Appendix B.*

A3. Apply SAD evaluation (3-16) to shortlist M candidates with the smallest residue energies (including MPM).

A4. Select an optimal mode that minimizes the Lagrangian cost from the short-listed candidates.

The proposed fast algorithm employs the inherent frequency characteristic of an original block and its predicted block without any a priori knowledge, such as a predefined threshold. This feature is considered one of the main advantages of the proposed algorithm in that it is appropriate for all sequences, without modification. Furthermore, the utilisation of matrices, $\tilde{\mathbf{\Omega}}_{(0,0)}$ and $\tilde{\mathbf{\Omega}}_{(m,n)}$ obtains the required frequency coefficients without an increase in the computational overhead.

3.3 Simulations, Comparisons, and Discussions

The simulation results are described in two subsections. The first subsection demonstrates the results of the proposed *Fintra* algorithm with various M settings, i.e., the number of shortlisted candidates for Lagrangian examination. Subsequently, a comparison is made with Pan et al's algorithm recorded in [64]. All the results are presented as improvements over the standard H.264/AVC benchmark, JM6.1e [56]. The test sequences of QCIF resolution (176x144) are classified into three different classes. The other settings are as follows:

1) All the frames are intra-coded as I-frames.
2) In total 50 frames of each sequence are processed.
3) Lagrangian rate-distortion optimisation is employed for mode selection.
4) Five different compression levels, i.e., Qp = 20, 24, 28, 32, 34, are considered.
5) Context-based Adaptive Binary Arithmetic Coding (CABAC) is used for entropy coding.

Finally, comparisons are given as the PSNR value of the luminance component, Y-PSNR (measured in dB), bit rate (in Kbits/sec.), and time reduction (computational performance). The equation for reduction in computation time is defined in (2-29).

3.3.1 Simulation results for various values of M

Table 3-1 summarises the performance of the proposed *Fintra* algorithm with various M options, i.e., M = 2, 3, and 4. Comparisons with the JM6.1e benchmark are given for Y-PSNR difference, bit-rate difference, and reduction in computational requirement. Note that each table entry is presented as an average outcome over five compression levels, i.e., Qp = 20, 24, 28, 32, 34.

General trends are identified as follows: the quality degradation is affected by the number of M selected. A better performance is expected as the number of shortlisted candidates is increased. For example, PSNR loss of between 0.04dB and 0.11dB were obtained with a shortlist of 4 candidates, compared to 0.12dB-0.32dB with M = 2. Similarly, bit rate achieved also responded to the variable, M settings. However, most of the test sequences encoded by the algorithms are still within an acceptable bit-rate difference of less than 3%.

With regard to computational efficiency, a constant time reduction is observed from one sequence to another for the algorithms with each M setting. That is attributed to the same coding method applied to the macroblock regardless of the texture composition. On average, 47%, 56%, and 65% computational savings are achieved by the *Fintra* algorithm with M = 4, 3, and 2, respectively.

Fig. 3-3 and Fig. 3-4 show PSNR-rate relationship diagrams for the *Mobile & Calendar* and *Stefan* sequences selected from Class C. The curve with the 'X' marker represents the performance of the JM6.1e benchmark. It also reflects an upper bound for the other three fast algorithms, i.e., M = 2 ('□' marker), M = 3 ('O' marker), and M = 4 ('Δ' marker). As expected, the curve with M = 4 demonstrates the best coding gain. However, it is noticed that a further increase in the number of candidates does not necessarily provide a significant improvement in overall performance. This observation is evident for all test sequences where a marginal difference is obtained between M = 3 and M = 4. Thus, we conclude that the proposed *Fintra* algorithm maintains an acceptable level of degradation in terms of rate and distortion provided that a minimum of 3 candidates are selected for Lagrangian examination.

Table 3-1
The average Y-PSNR difference, bit-rate difference, and time saving for the *Fintra* algorithms with different *M* settings.

Sequences	Y-PSNR Gain (dB)			Bit rate Difference (%)			Time Saving cf. JM6.1e (%)		
	M=4	M=3	M=2	M=4	M=3	M=2	M = 4	M = 3	M = 2
Class A									
Akiyo	-0.05	-0.06	-0.14	1.08%	1.29%	2.39%	50.60%	56.66%	65.13%
Container	-0.05	-0.08	-0.12	2.41%	2.59%	3.10%	46.94%	55.79%	65.29%
Class B									
Silent Voice	-0.04	-0.07	-0.15	1.28%	1.68%	3.39%	46.81%	55.45%	64.84%
News	-0.08	-0.11	-0.17	0.80%	1.10%	2.11%	47.54%	56.50%	65.29%
Class C									
Fun Fair	-0.07	-0.12	-0.20	0.12%	0.36%	1.41%	48.21%	57.02%	65.84%
Mobile & Calendar	-0.11	-0.16	-0.28	-0.03%	0.10%	0.82%	47.54%	56.46%	65.19%
Stefan	-0.13	-0.20	-0.32	0.01%	0.07%	0.66%	47.87%	57.54%	66.11%
Train and Tunnel	-0.07	-0.11	-0.19	0.33%	0.56%	1.55%	43.99%	56.69%	65.32%
Averages	**-0.07**	**-0.11**	**-0.19**	**0.75%**	**0.96%**	**1.92%**	**47.43%**	**56.51%**	**65.37%**

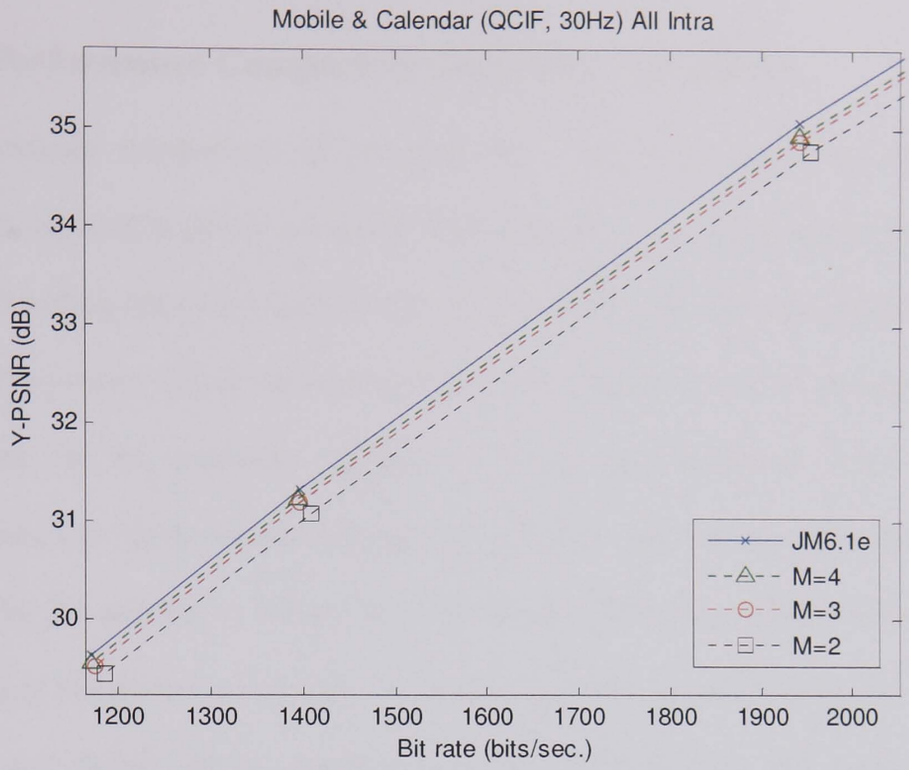


Fig 3-3 The PSNR-rate distortion curves for the *Mobile & Calendar* sequence (Class C) obtained by the JM6.1e algorithm and the proposed *Fintra* algorithm with different *M* settings.

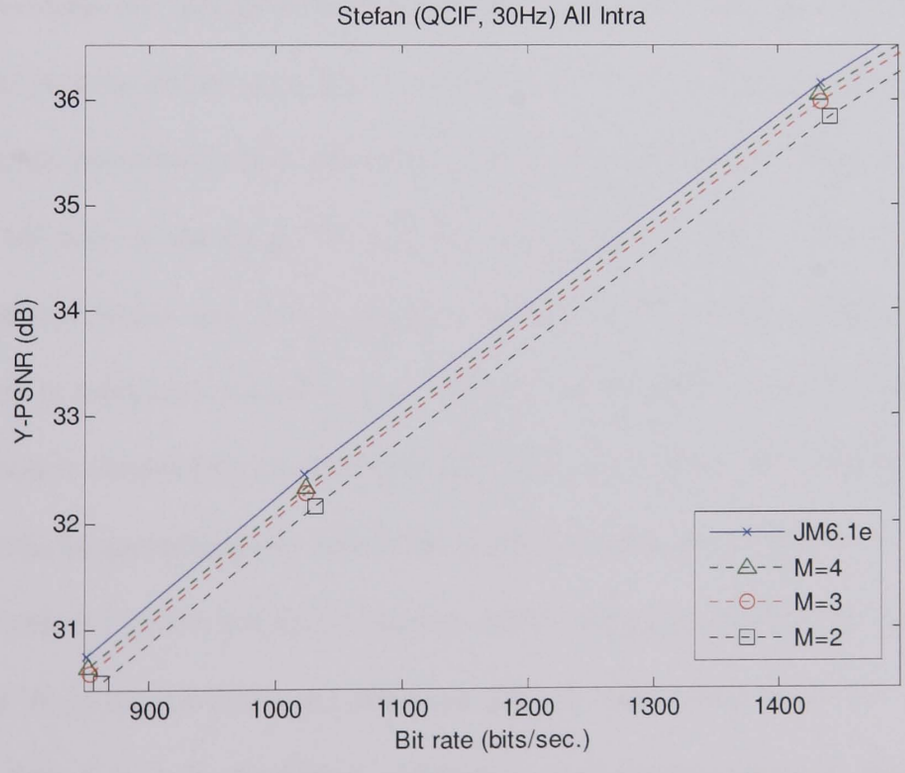


Fig 3-4 The PSNR-rate distortion curves for the *Stefan* sequence (Class C) obtained by the JM6.1e algorithm and the proposed *Fintra* algorithm with different *M* settings.

3.3.2 Performance Comparison with other algorithms

A performance comparison of the proposed *Fintra* algorithm with Pan et al's algorithm reported in [64] is presented. Both algorithms select 4 candidates ($M = 4$) in the *I4MB* coding option for intra-frames. Note that the other fast algorithms applied to the *I16MB* coding option mentioned in [64] are excluded in this simulation. Apart from that, all the parameter settings for these two algorithms were identical. Comparisons are given for rate-distortion performance and computational speedup.

Fig 3-5 and Fig. 3-6 show two PSNR-rate relationship diagrams for the most challenging sequences in Class C, i.e., *Mobile & Calendar* and *Stefan*. The solid line, dashed, and dotted curves represent the JM6.1e benchmark, the proposed Fintra algorithm, and Pan et al's method, respectively. Generally, the PSNR-rate curves of both algorithms are virtually identical to that provided by the JM6.1e software. A magnified section of each of the curves is provided for both figures. It is observed that the proposed algorithm reduces the average PSNR by less than 0.15dB (see Table 3-1) for any bit rate in the range. The PSNR degradation of Pan et al's algorithm is approximately twice that of the proposed method. It is worth noting that human perception is especially sensitive at a PSNR of less than 36dB. Unnoticeable picture degradation is obtained by the proposed algorithm, as the average PSNR difference is less than the human perception threshold, widely recognised as 0.20dB.

Table 3-2 shows the time reduction achieved by both algorithms. Note that the results of the proposed Fintra algorithm are directly taken from Table 3-1 where $M = 4$. It is clear that both algorithms maintain a constant reduction in computational requirement regardless of the class of sequence. A minor variation in time reduction between 44% and 50% is reported for the proposed *Fintra* algorithm, compared to

Table 3-2
The average time saving achieved by the *Fintra* algorithm and Pan et al's algorithm.

Sequences	Time Saving cf. JM6.1e (%)	
	Fintra Alg.	Pan et al's Alg.
Class A		
Akiyo	50.60%	41.68%
Container	46.94%	40.69%
Class B		
Silent Voice	46.81%	42.17%
News	47.54%	40.87%
Class C		
Fun Fair	48.21%	41.28%
Mobile & Calendar	47.54%	41.53%
Stefan	47.87%	38.87%
Train and Tunnel	43.99%	44.87%
Averages	**47.44%**	**41.50%**

38% – 44% for Pan et al's algorithm. Thus, we conclude that the proposed *Fintra* algorithm outperforms the algorithm reported in [64] in all respects.

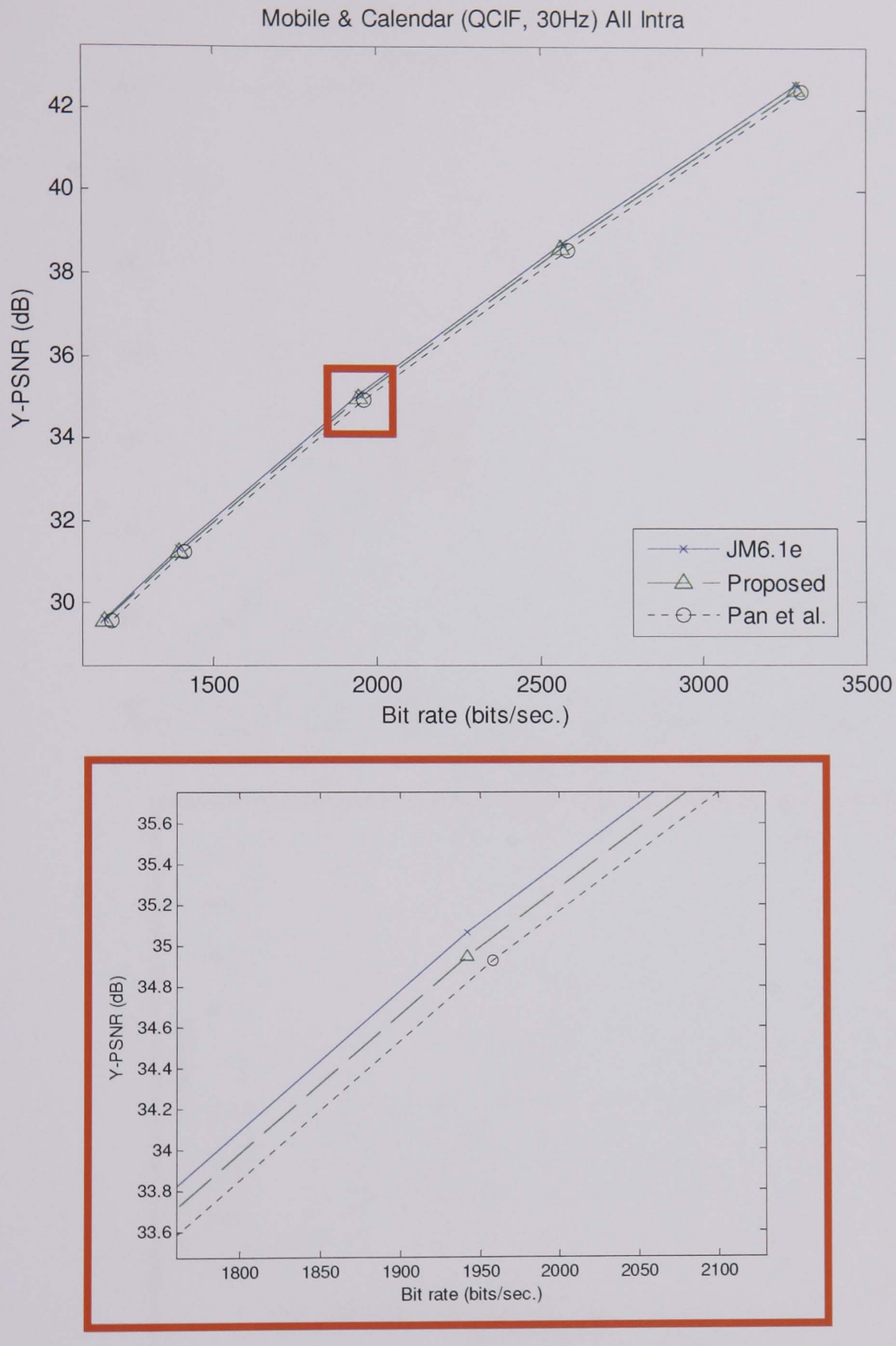


Fig 3-5 Performance comparison of the proposed *Fintra* algorithm and Pan et al's algorithm employing *Mobile & Calendar* sequence (Class C) in terms of rate-distortion (above); Magnified section of the curves (below).

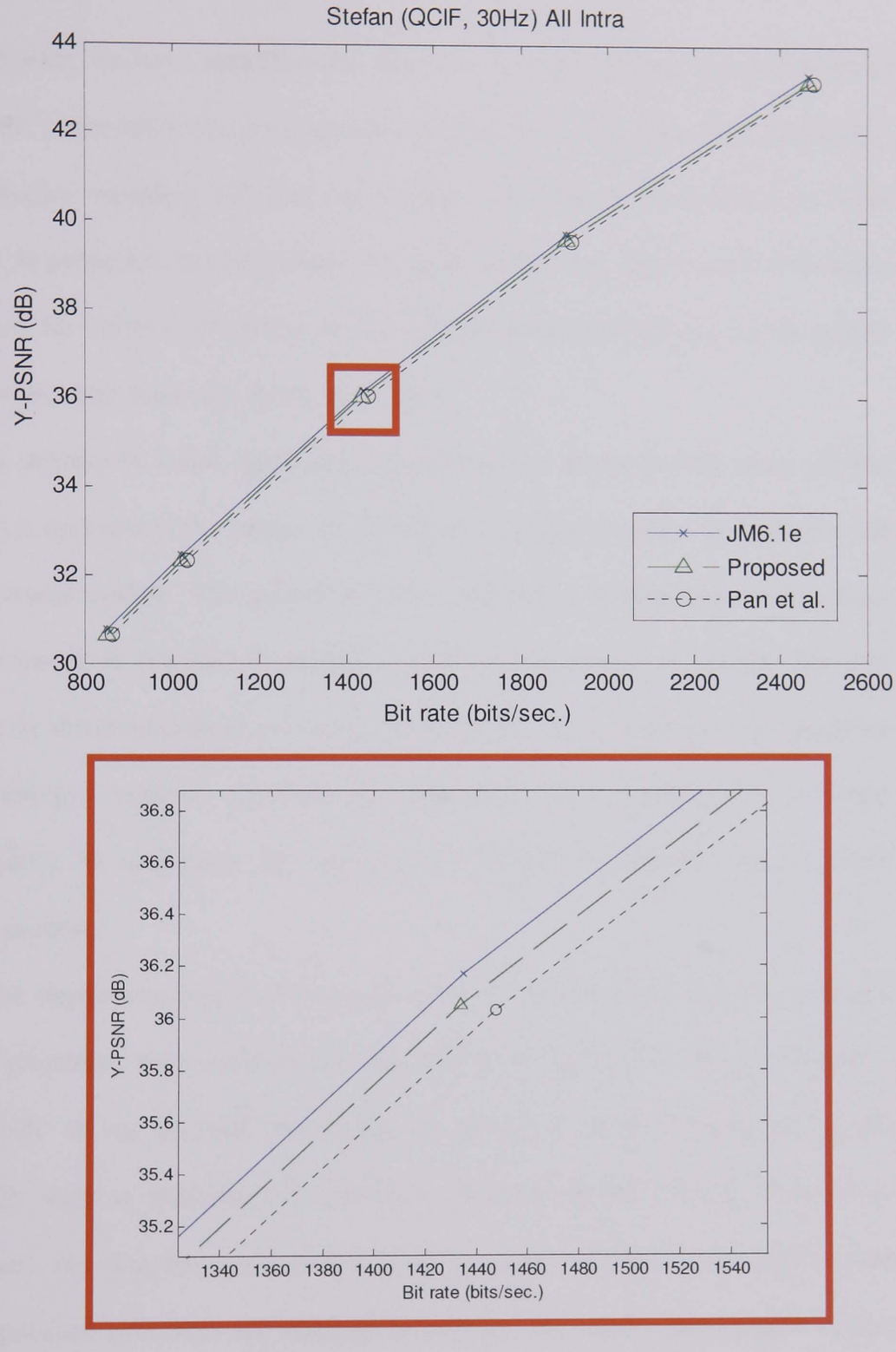


Fig 3-6 Performance comparison of the proposed *Fintra* algorithm and Pan et al's algorithm employing *Stefan* sequence (Class C) in terms of rate-distortion (above); Magnified section of the curves (below).

3.4 Summary

In this chapter, we have identified the difficulty in implementing a software-based H.264/AVC coder due to the complex encoder design. To overcome this technicality, several studies regarding efficient computation for intra-frame coding have been proposed. In particular, spatial domain approaches have been widely used. Alternative approaches, for instance, prediction in a temporal fashion and utilising a joint spatial-frequency decision, were also briefly described.

In contrast to other approaches detailed in the literature, the proposed fast algorithm is operated in the Integer DCT (IntDCT) domain to select an optimal mode for intra-frame coding. The proposed *Fintra* algorithm intelligently selects fewer candidate members required to undergo expensive Lagrangian evaluation. The core technique is the employment of matrix formulae for the computation of minimum residue energy, without significantly increasing the computational overhead. Subsequently, M candidates are acquired for further examination in the mode selection process.

The implementation of the proposed *Fintra* algorithm also results in several plausible properties. First and foremost, the algorithm employs the inherent frequency characteristic of an original block and its predicted block without any priori knowledge, such as predefined threshold or other priori macroblock information. Furthermore, the complete computation of all frequency spectra is not required, and the extrapolation processes are only performed for the modes that require further Lagrangian examination. Finally, the matrix formulae can be calculated (as shown in Appendix B) and stored in advance.

Performance comparison of the algorithms with different numbers of shortlisted candidates was made in Section 3.3.1. The results determine the minimum

value of M should be 3 in order to maintain a better compromise between rate and distortion performance. An extensive comparison with other algorithms detailed in the literature was also conducted to verify the efficiency of the proposed method. The rate-distortion curves show that the proposed algorithm achieves the same coding performance in terms of picture quality and compression ratio as that of the H.264/AVC standard, yet reduces the computational requirement by up to 57%.

Chapter 4
Multiple Mode Selection for Inter-frame Coding

To complete the hybrid structure for a fast H.264/AVC implementation, this chapter further discusses two proposed methods to achieve efficient inter-frame coding. As mentioned in Chapter 2, the coding for an inter-frame entails a mode selection process, which represents a composition of motion search and rate-distortion optimisation. The computation of the Lagrangian cost for each mode necessitates completion of the encoding-decoding cycle. So far, the H.264/AVC algorithm incorporates a brute force algorithm to determine the mode which produces the least Lagrangian outcome. Consequently, the computational burden introduced by the mode selection process is far more significant than that of any other video coding standard.

Several studies have been conducted to alleviate the computational burden imposed by the brute force algorithm. Basically, there are two search strategies: 'SKIP-mode detection' and 'structure incorporating hierarchical decision'. Two proposed inter-mode selection algorithms, namely, *Finter1* and *Finter2*, are presented. The success of the two proposed algorithms is achieved by structures comprising several hierarchical decisions. They aim at providing different levels of early termination for mode selection

to avoid the full cycle of Lagrangian examination. The descriptions of the *Finter1* and *Finter2* are recorded in a parallel way for comparison purposes.

The results of simulations employing both algorithms are summarised in Section 4.4. The performance comparisons are given as PSNR difference, bit-rate difference, and reduction in computational time. Extensive simulations such as rate-distortion diagrams are also provided prior to the overall conclusions discussed in the last section.

4.1 The Existing Solutions For Fast Inter-frame Coding

The mode selection process in inter-frame coding requires a significant amount of computation in the encoder. This has been the subject of several studies. Generally, most fast inter-mode selection approaches detailed in the literature can be classified as 'SKIP-mode detection' or 'using a hierarchical decision'. The former strategy specifically targets those macroblocks encoded with the *SKIP* mode (denoted skipped macroblocks). These algorithms provide early termination of the mode selection process for skipped macroblocks without the involvement of Lagrangian evaluation. This is the simplest approach. In contrast, hierarchical structures pre-define levels as collections of modes comprising different macroblock types. Decisions are then made at each level. The next two subsections review existing methods that follow each of these strategies.

Table 4-1
Average proportion of skipped macroblocks in inter-frame coding with respect to compression levels (indicated by *Qp*) and image motion activity (categorised by class).

		Low ← Compression Levels → High				
Classes and Sequences		*Qp* = 20	*Qp* = 24	*Qp* = 28	*Qp* = 32	*Qp* = 36
Class A	*Akiyo*	79.40%	83.14%	87.13%	90.79%	94.39%
	Container	51.42%	69.54%	80.62%	87.52%	93.35%
Class B	*News*	67.42%	73.25%	78.03%	82.74%	87.23%
	Silent Voice	61.06%	68.10%	73.43%	78.46%	83.53%
Class C	*Mobile & Cal.*	0.25%	0.76%	2.80%	11.08%	27.83%
	Fun Fair	1.73%	2.55%	4.21%	7.98%	15.07%

4.1.1 Early Detection for Skipped Macroblocks

Grecos and Yang [78] achieved early prediction of skipped macroblocks by utilising a set of smoothness constraints, considered as an improvement of the conditions outlined in [69, 70, 33]. Kim et al. [71, 72] proposed a threshold method derived from the definition of the integer DCT transform and quantisation process to obtain an all zero DCT-block. Kannangara et al. [73] used an estimated rate-distortion model for the Lagrangian multiplier parameter, thus coding without rate-distortion constraints to achieve early SKIP block detection.

The appeal of SKIP-block detection is that it is simply computed. However, only limited speed-up is achieved depending on the contents of the video signal and the selected compression level. Table 4-1 demonstrates this in terms of the proportion of skipped macroblocks for different classes of motion activity. The table also illustrates the correlation between the proportion of skipped macroblocks and the quantisation factor, *Qp*. It is clear that the success of fast inter-mode selection algorithms that

consider skipped macroblocks only is restricted to video signals comprising low motion activity or when high levels of compression are required.

4.1.2 Fast Algorithms Featuring Hierarchical Decisions

Fast algorithms that consider not only skipped macroblocks but also other features are able to demonstrate improved performance for a wider range of video signals. This is attributed to a detailed analysis at each of a number of levels. Macroblocks can be associated with different categories adaptable to compression levels. In theory, hierarchical algorithms outperform the simple SKIP-detection strategy in terms of prediction accuracy and coding efficiency. In [74], Jing and Chau employ an inequality to classify macroblocks into two groups. Each group is encoded with a pre-defined collection of modes. Similarly, Yang and Wang [75] advocated a fuzzy classifier to categorise modes into three motion groups, simple motion, complex motion and the INTRA motion group. Wu et al [76, 77] integrated their algorithm with a Sobel operator to determine the homogeneity of a macroblock. Macroblocks that belong to a particular type of homogeneity are encoded with a specific combination of modes. The aforementioned algorithms demonstrated good coding performance. However none of them identified skipped-macroblocks independently and each had to undergo at least one Lagrangian evaluation.

4.2 The Proposed *Finter1* and *Finter2* Algorithms

Success of two proposed fast mode selection algorithms, *Finter1* and *Finter2*, for inter-frame coding is achieved by discarding the least possible block size. Mode knowledge of the previously encoded frame(s) is employed by the proposed *Finter1* algorithm, whereas the *Finter2* algorithm incorporates temporal similarity detection and the detection of different moving features within a macroblock. However, both *Finter1* and *Finter2* make use of a general tendency: a mode having a smaller partition size may be beneficial for detailed areas during the motion estimation process, whereas a larger partition size is more suitable for homogeneous areas [83, 74, 75, 77]. The reasons given are that: (a) homogeneous macroblocks tend to contain fewer moving features requiring multiple motion descriptors, and (b) owing to the homogeneous content, the distortion costs arising from incorrect predictions are often insignificant. Consequently, a spatial complexity measurement is developed to determine the content of the macroblocks being considered

4.2.1 Algorithm Formulation

We derive a low-cost complexity measurement based on summing the total energy of the AC coefficients to estimate the block detail. The AC coefficients are obtained from the high frequency coefficients of each $M \times N$ block:

$$E_{AC} = \left(\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} W^2(u,v) \right) - \text{DC energy} \qquad (4\text{-}1)$$

From (4-1), the total energy of the AC components, E_{AC}, of an $M \times N$ block is the sum of all the DCT coefficients, $W(u,v)$, except for the DC location where $u = v = 0$. However, the computation for $W(u,v)$ necessitates the need for forward DCT transformation defined in (2-1). Due to the orthonormality property, the accumulated

energy of the DCT coefficients can be represented by the total energy of an $M \times N$ block in the pixel domain. Thus, (4-1) can be further simplified as

$$E_{AC} = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left(w^2(m,n)\right) - \frac{1}{M \times N}\left(\sum_{n=0}^{N-1}\sum_{m=0}^{M-1} w(m,n)\right)^2 \qquad (4\text{-}2)$$

where $w(m,n)$ denotes the pixel sample in an $M \times N$ block. The first term in (4-2) accounts for the total energy of the pixel intensities within an $M \times N$ block, and the second term reflects the DC energy, which can be expressed by the mean square intensity of an $M \times N$ block. (4-2) further indicates that the energy of the AC components of a macroblock can be represented by the variance.

Since variable block decompositions are available in the inter-modes, three testing sizes are selected for evaluation. They are the partitioned blocks of size 16x16, 8x8, and 4x4 pixels. The complexity measurements in (4-2) for each testing size need to be made. In the worst case, the measurements are repeated for up to 21 times for the same macroblock, i.e., one 16x16 macroblock + four decomposed 8x8 blocks + sixteen 4x4 subblocks. Thus, (4-2) is rewritten as forming three piecewise equations to reduce the computational redundancy.

$$E_{AC} = \begin{cases} \sum_{x=1}^{16} E_x - \left(\frac{1}{16}\sum_{x=1}^{16} S_x\right)^2 & (4\text{-}3a) \\ \sum_{x=1}^{4} E_{4(y-1)+x} - \left(\frac{1}{8}\sum_{x=1}^{16} S_{4(y-1)+x}\right)^2, \; y=\{1,\cdots,4\} & (4\text{-}3b) \\ E_x - \left(\frac{1}{4} S_x\right)^2, \; x=\{1,\cdots,16\} & (4\text{-}3c) \end{cases}$$

where $E_n=\{e_1, e_2, \ldots, e_{16}\}$ and $S_n=\{s_1, s_2, \ldots, s_{16}\}$ represent the sum of energies and intensities of the 4×4 blocks decomposed from a macroblock respectively, with the scanning pattern shown in Fig. 4-1. The first piecewise equation is applied to a

macroblock with block size of 16×16 pixels; the second is for 4 blocks, $y = \{1, 2, 3, 4\}$ of 8×8 pixels; and the last is applicable to the 16 decomposed 4×4 blocks.

Evaluating the maximum sum of the AC components is the next target. By definition, the largest variance is obtained from the block comprising a checkerboard pattern in which every adjacent pixel is the permissible maximum ($w_{\max}$) and minimum ($w_{\min}$) value alternately. Thus, $E_{\max}$, the maximum sum of the AC components of an $M \times N$ block is

$$E_{\max} = \frac{MN}{2}\left[\left(w_{\max}^2 + w_{\min}^2\right) - \frac{1}{2}\left(w_{\max} + w_{\min}\right)^2\right]. \tag{4-4}$$

Note that $E_{\max}$ can be calculated in advance. Then the criterion to assess the complexity R_B of a macroblock MB is

$$R_B = \frac{\ln(E_{AC})}{\ln(E_{\max})} \tag{4-5}$$

The function of the natural logarithm is to linearise both $E_{\max}$ and E_{AC} such that the range of R_B can be uniformly split into 10 subgroups. According to experiments, a macroblock with $R_B > 0.70$ is considered to be a high-detailed block. In our evaluation, we select $R_B = 0.75$.

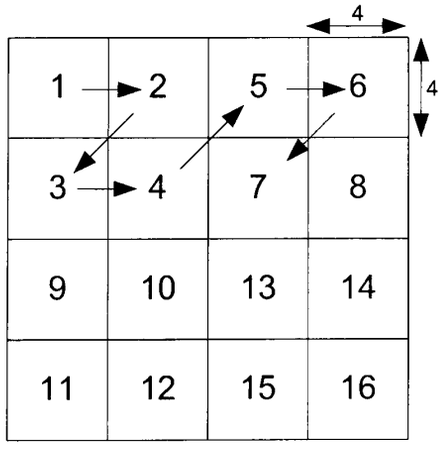


Fig. 4-1 Proposed scanning order for E_n and S_n, the energy and sum of intensities in a 4×4 block in order to reduce computational redundancy.

4.2.2 The *Finter1* Algorithm

Fig. 4-2 shows the flowchart of the proposed *Finter1* algorithm that incorporates the complexity measurement. In total, 7 partition sizes are recommended by H.264 for P-frames, namely: 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4, as well as *SKIP*, and other two intra-extrapolation options, *I4MB* and *16MB*. However, in our complexity measurement, only 3 categories, of sizes 16×16, 8×8, and 4×4, are selected as test block sizes. We denote them as *Cat0*, *Cat1*, and *Cat2*, respectively.

The proposed *Finter1* algorithm provides a recursive way to determine the complexity of each macroblock. Firstly, a macroblock of 16×16 pixels is examined with (4-3a). A *Cat0* tag is given if it is recognized as being a homogenous macroblock. Otherwise, the macroblock is decomposed into 4 blocks of 8×8 pixels. Note that an 8×8 block is recognized as high-detailed if it satisfies two conditions:

(a) The R_B in (4-5) is greater than 0.75, and it is decomposed into four 4×4 blocks;

(b) One of its four decomposed 4×4 blocks is high-detailed as well.

If an 8×8 block satisfies the first condition but not the second, it is still recognized as low-detailed. After checking all the 8×8 blocks, a *Cat2* tag is given to a macroblock which possesses more than two high-detailed blocks, otherwise a *Cat1* tag is assigned. Table 4-2 displays the relationship between the three categories and the nine members of the inter-frame modes. It is observed that the *Cat0* category covers the least number of members of the inter-frame mode, whereas the *Cat2* category contains all the available members. The table further indicates that the higher detailed the macroblocks are, the more prediction modes the proposed algorithm has to check.

Table 4-2
Relationship between the three categories in the proposed algorithm and the 9 inter-predicted modes.

Category	Corresponding Modes
Cat0	16x16, *SKIP*, *I16MB*, *I4MB*
Cat1	16x16, 16x8, 8x16, 8x8, *SKIP*, *I16MB*, *I4MB*
Cat2	16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4, *SKIP*, *I16MB*, *I4MB*

Mode Knowledge of Previously Encoded Frame(s)

A trade-off between efficiency and prediction accuracy exists. If a *Cat2* category is assigned less often, the efficiency of the algorithm will increase, but the chance of erroneous prediction also increases. An improved method is proposed, that considers the mode knowledge at the same location in the previously encoded frame. Since most of the macroblocks are correlated temporally, it is easy to see that the mode decision in the previous frame contributes reliable information for revising the erroneous prediction that may be indicated by its intrinsic complexity information. Therefore, the suggestion is first to convert all the mode decisions in the previous frame into the corresponding categories. Then, the prediction is revised to the higher category if that of the corresponding historic data is higher than the current predictor. However, no action is taken if the reverse situation is true.

The Algorithm of *Finter1*:

Cat0 category algorithm:

A1 Obtain a motion vector for a 16×16 macroblock by using the full search algorithm with search range of ±8 *pixels.*

A2 Compute the Lagrangian costs of SKIP, two intra-coding options, and the activated partition sizes *to find a final mode decision for the current macroblock.*

Cat1 category algorithm:

B1 Obtain a motion vector for each of the four 8×8 blocks in a macroblock by using the full search algorithm with search range of ±8 *pixels.*

B2 Continue to search for motion vector(s) for the 8×16 blocks, 16×8 blocks, and 16×16 macroblocks by referring only to the 4 search points, i.e., the motion vectors of the four 8×8 blocks.

B3 Perform step A2 to find the final mode decision for the current macroblock.

Cat2 category algorithm:

C1 Obtain a motion vector for each of the sixteen 4×4 blocks in a macroblock by using the full search algorithm with search range of ±8 *pixels.*

C2 Continue to search for motion vector(s) for 8×4 blocks, 4×8 blocks, and 8×8 blocks by referring only to the 16 search points, i.e., the motion vectors of the sixteen 4×4 blocks.

C3 Perform the steps B2 to B3 to find the final mode decision for the current macroblock.

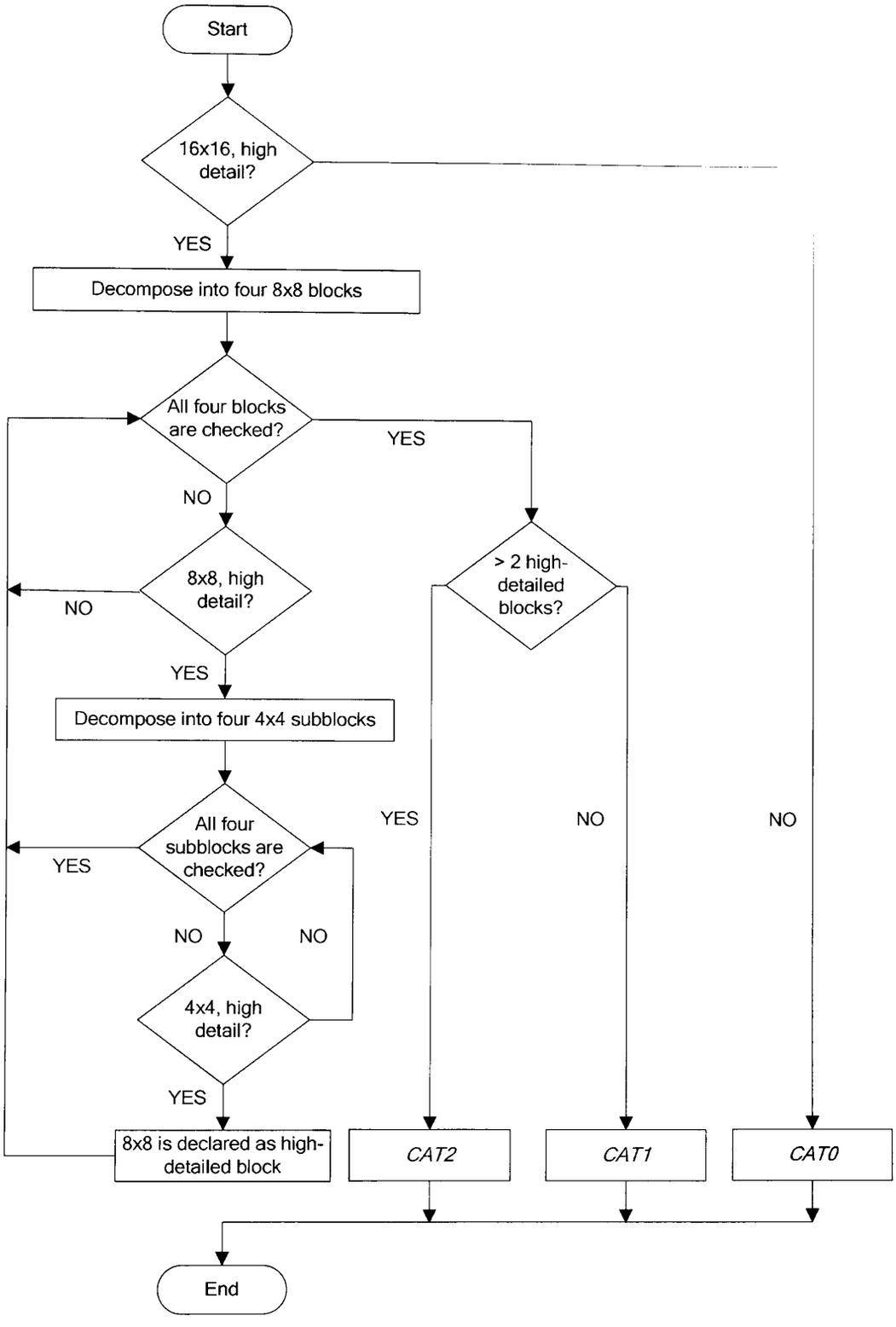


Fig. 4-2 Flowchart of the proposed *Finter1* algorithm incorporating the complexity measurement for a macroblock. A macroblock will be recognized as one of three categories, namely *Cat0*, *Cat1* and *Cat2*.

4.2.3 The Finter2 Algorithm

The efficiency of the proposed *Finter2* is achieved by introducing two additional measurements targeted at two kinds of encoded macroblocks: (a) macroblocks encoded with *SKIP* mode (direct copy from the collocated macroblock); (b) macroblocks encoded by the inter-frame modes with larger decomposed partition size (greater than 8×8 pixels). By successfully identifying these two kinds of macroblocks, the encoder is not required to examine them with all possible inter-frame modes, which saves encoding time.

Measurement of Temporal Similarity

The proposed algorithm distinguishes the macroblocks encoded with SKIP (denoted as skipped macroblocks). The skipped macroblocks can be found in inter-frames where the pixel information is almost identical to that of the collocated macroblock. Thus, they can be detected by means of the temporal similarity between the two macroblocks. As skipped macroblocks tend to occur in clusters, such as in a patch of static background, we suggest that the current macroblock undergoes temporal similarity detection if at least one of the skipped macroblocks is found in the neighbours, depicted in Fig. 4-3. The decision for temporal similarity detection is defined as

$$\text{Decision} = \begin{cases} 1, \text{SAD(residue)} < Thd_A \\ 0, \text{otherwise} \end{cases} \tag{4-6}$$

$$\text{SAD(residue)} = \sum_{m=0}^{15}\sum_{n=0}^{15} \left| w_t(m,n) - \hat{w}_{t-1}(m,n) \right| \tag{4-7}$$

where $w_t(m,n)$ and $\hat{w}_{t-1}(m,n)$ represent the current macroblock and predicted macroblock taken directly from the collocated macroblock in the previous frame,

respectively. Thd_A is an adaptive spatially varying threshold. Its value is dependent on the outcome of the neighbouring macroblocks.

$$Thd_A = \min(S_{N1}, S_{N2}, S_{N3}, S_{N4}) \tag{4-8}$$

where S_{N1}, S_{N2}, S_{N3}, and S_{N4} are the SAD values of the four nearest encoded neighbours, N_1, N_2, N_3, N_4, in Fig. 4-3, respectively. They are valid and pre-stored in the system if and only if their corresponding macroblocks are skipped macroblocks. A non-zero outcome in (4-6) indicates that the currently processed block is a skipped macroblock.

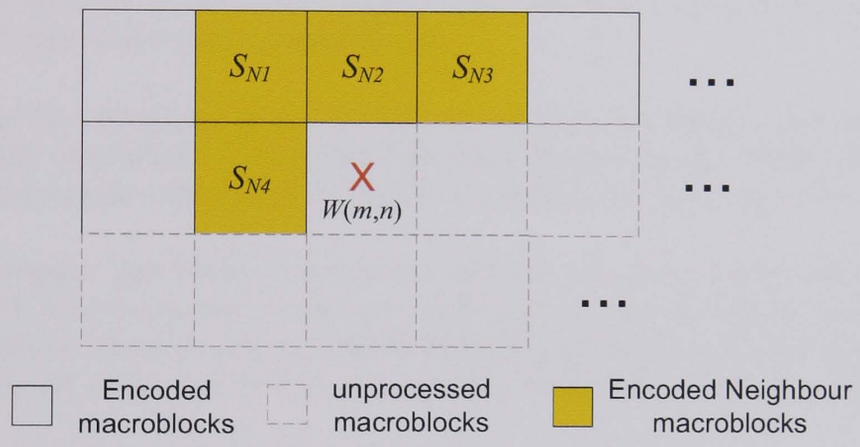


Fig. 4-3 The relative position of four nearest encoded neighbours with respect to the currently processed macroblock.

Measurement of Block-Based Motion Consistency:

The tendency that inter-frame modes with larger partition size (of sizes 8×8, 8×16, 16×8, and 16×16 pixels) are more suitable to encode homogeneous macroblocks has been verified by a number of authors [74, 75, 77, 83]. By contrast, macroblocks containing moving features appear more detailed and therefore require use of smaller block sizes. Thus, the proposed algorithm suggests checking the motion vector of each 8×8 block decomposed from a highly detailed macroblock. If consistency among the motion vectors (of four 8×8 blocks) within a highly detailed macroblock exists,

the proposed algorithm checks the inter-modes with partition size greater than 8×8. If the motion vectors are not consistent, all possible inter-frame modes are searched.

The Algorithm, *Finter2*:

Fig. 4-4 shows a flowchart of the proposed *Finter2* algorithm summarised as follows:

D 1 *Turn off all flags including SKIP, INTRA and all inter-modes*

D 2 *Check if one of the four nearest neighbours of the current macroblock is a skipped macroblock. Implement STEP 3 if the situation is true. If not, go to D4.*

D3 *Obtain a threshold, Thd_A, from (4-8). Compare Thd_A with the sum of absolute difference between the current macroblock and the previous macroblock at the same position. If the sum is smaller than the threshold, turn on the flag for SKIP only. Otherwise, continue to D4.*

D4 *Check the complexity of the macroblock using equation (4-3a) and (4-5). If the current macroblock is homogeneous, turn on the flag for I4MB, I16MB and the inter-mode with partition size 16×16. Otherwise, continue to D5.*

D5 *Decompose the highly detailed macroblock into four non-overlapping 8×8 blocks. Check whether the motion vectors of the four blocks are consistent. If consistent, check the flags of I4MB, I16MB, and the inter-modes with partition size 8×16, 16×8, and 16×16 and go to D7. Otherwise, continue to D6.*

D6 *Turn on all flags and use sixteen motion vectors obtained from inter-modes with partition size 4×4 as searching points for the inter-mode with partition size 4×8 and 8×4 rather than performing full search. Then, continue to D7.*

D7 *Utilise the four motion vectors obtained from four 8×8 blocks as searching points for the inter-modes with partition size 8×16, 16×8, and 16×16 rather than performing full search.*

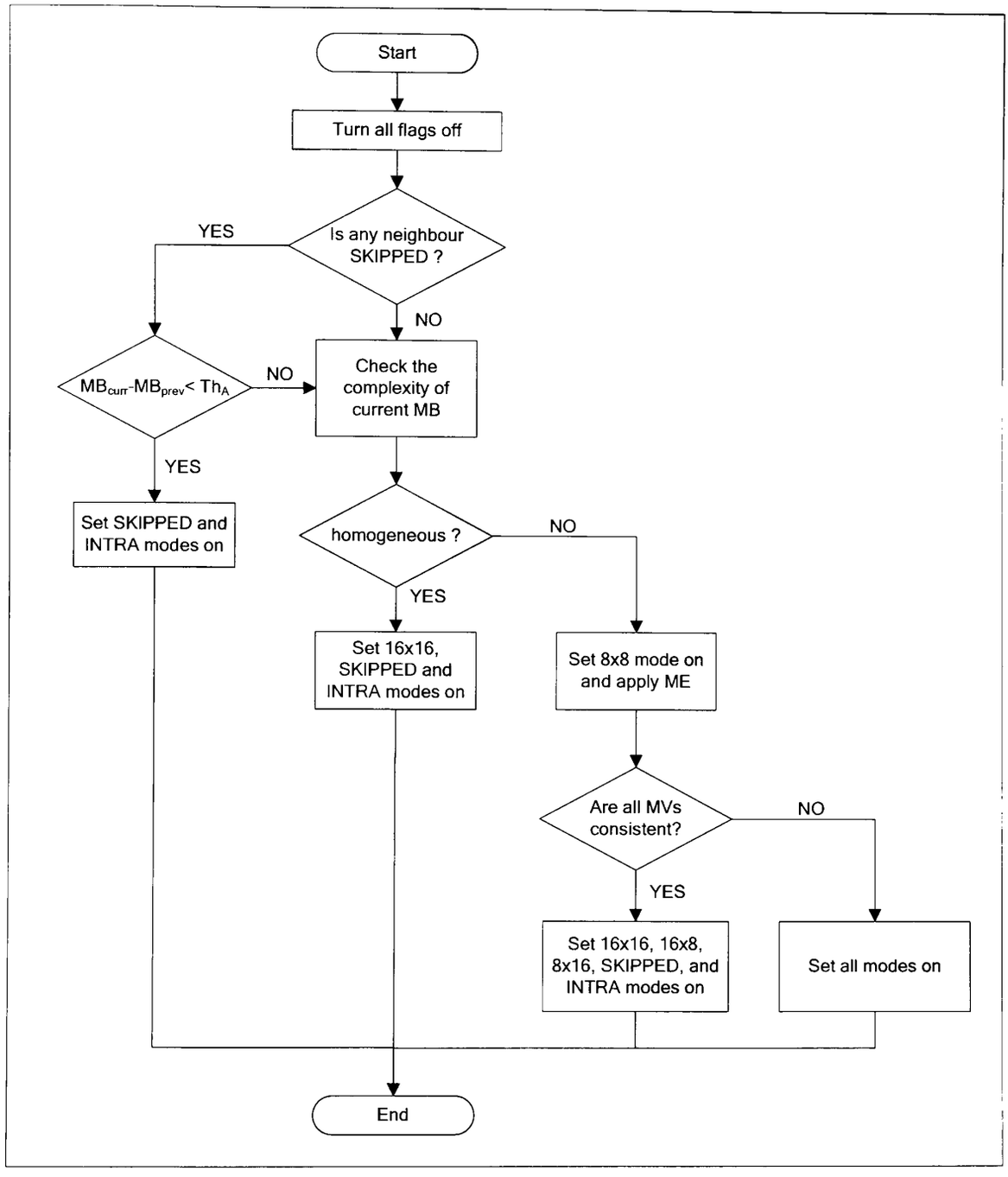


Fig. 4-4 Flowchart of the proposed *Finter2* algorithm incorporating the complexity measurement for a macroblock, temporal similarity, and the detection of different moving features within a macroblock.

4.3 Simulations, Comparisons, and Discussions

This section compares the results of the *Finter1* and *Finter2* algorithms. The testing benchmark was the JM6.1e software version provided by the Joint Video Team (JVT). The selected sequences in two different resolutions, namely, QCIF (144 x 176) and CIF (288 x 352) formats, are classified into three different classes. The other settings are given as follows:

1) Lagrangian Rate-Distortion Optimisation (RDO) is employed for mode selection.
2) Only one reference frame is used for prediction.
3) In total 50 frames of each sequence are processed.
4) Five different compression levels, i.e., Qp = 20, 24, 28, 32, and 34, are used.
5) Intra-coding is performed for the first frame only.
6) The precision of the motion vector is ¼ pixel.
7) Context-based Adaptive Binary Arithmetic (CABAC) is employed for coding.

Finally, comparisons are give as the PSNR value of the luminance component, Y-PSNR (measured in dB), bit rate (described in Kbits/sec), and reductions in computational requirement defined in (2-29).

4.3.1 Picture degradation, compression ratio, and speedup

Table 4-3 summarises the performance of the two proposed algorithms, *Finter1* and *Finter2*. Note that each table entry is presented as an average outcome from the five compression levels. The general trends are identified as follows: the quality degradation of most sequences for both algorithms is still maintained within an acceptable level of 0.20dB. It is noticed that the *Finter2* algorithm performs better than the *Finter1* method, in all respects. However, a marginal PSNR difference of at

most 0.07dB is obtained when a comparison is made between the two algorithms. In contrast, the tendency of bit rate increase is directly proportional to the class of sequence. The test sequences of Class A attain the least bit rate increase, whereas the high motion sequences in Class B and Class C produce higher bit rates than those of the H.264 standard. This situation is especially obvious for some Class C sequences, for instance 10.64% and 15.38% obtained from *Stefan* and *Train & Tunnel*, respectively, when employing the *Finter2* coding scheme.

With regard to computational reduction, it is correlated to the sequence class. Note that a fast motion estimation algorithm recommended by the JM6.1e software has been incorporated in the simulation. On average, a saving of 47% in encoding time is achieved for the *Finter2* algorithm and 33% for the *Finter1* algorithm. It is interesting to observe that the *Finter2* results provide a significant speedup for Class A test sequences, while the performance drops (approximately 20%) when spatial correlation/motion of the test sequences increase. The explanation is that the skipped macroblocks, which benefit the coding of the proposed *Finter2* algorithm, are abundant in low motion sequences, while the high motion sequences require more predictions from previous frames rather than direct copy. Thus, the advantage of *Finter2* becomes less significant in Class B and Class C test sequences. In any case, the general speedup for both proposed algorithms is in excess of 15%.

Table 4-3
The Y-PSNR difference, bit-rate difference, and time saving of the *Finter1* and *Finter2* algorithms averaged for different compression ratios.

Sequences	Y-PSNR Gain (dB)		Bit Rate Difference (%)		Time Saving cf. JM6.1e (%)	
	Finter2	*Finter1*	*Finter2*	*Finter1*	*Finter2*	*Finter1*
Class A						
Akiyo	-0.10 dB	-0.12 dB	0.18%	2.09%	48.05%	33.06%
Container	-0.07 dB	-0.08 dB	-0.10%	4.41%	43.00%	31.81%
Hall Monitor	-0.05 dB	-0.08 dB	4.30%	7.86%	49.34%	35.44%
Class B						
Paris (CIF)	-0.11 dB	-0.15 dB	6.31%	7.28%	35.63%	29.00%
Silent Voice	-0.09 dB	-0.12 dB	5.79%	5.56%	40.17%	33.37%
News	-0.13 dB	-0.17 dB	4.38%	6.42%	39.94%	30.68%
Class C						
Fun Fair	-0.14 dB	-0.23 dB	8.73%	8.48%	19.00%	23.19%
Harbour (CIF)	-0.17 dB	-0.15 dB	2.43%	3.23%	17.01%	21.32%
Mobile & Calendar	-0.12 dB	-0.16 dB	0.05%	2.80%	15.05%	19.17%
Stefan	-0.17 dB	-0.13 dB	10.64%	6.03%	24.41%	22.22%
Tempete	-0.11 dB	-0.15 dB	0.85%	1.89%	19.86%	20.73%
Train & Tunnel	-0.08 dB	-0.11 dB	15.38%	9.78%	23.58%	19.21%
Waterfall (CIF)	-0.11 dB	-0.12 dB	1.39%	1.23%	34.32%	29.08%
Averages	**-0.11 dB**	**-0.12 dB**	**4.64%**	**5.16%**	**31.49%**	**26.79%**

4.3.2 Rate-distortion performance

We further discuss the performance of the two algorithms with the JM6.1e encoder for a wide range of compression levels. Fig. 4-5 shows the PNSR-rate relationship for six test sequences selected from the different classes. The curve with the 'X' marker represents the performance of the JM6.1e benchmark. It also reflects an upper bound for the fast algorithms, i.e., the *Finter1* algorithm ('Δ' marker) and the *Finter2* algorithm ('◊' marker).

Generally, the PSNR-rate performance of both proposed algorithms have attained more significant distortion at higher bit rates. It is also noted that the rate-distortion curves provided by the *Finter2* method show better performance than those of the *Finter1* algorithm at all bit rates. This is true in most cases except for the *Stefan* sequence where a PSNR degradation is observed for the *Finter2* algorithm. The reason is attributed to a significant increase in bit rate when the *Stefan* sequence is encoded with the *Finter2* algorithm, as shown in Table 4-3. However, without a loss of generality, we conclude that the *Finter2* algorithm provides better overall performance than the *Finter1* method for most of the sequences at all bit rates.

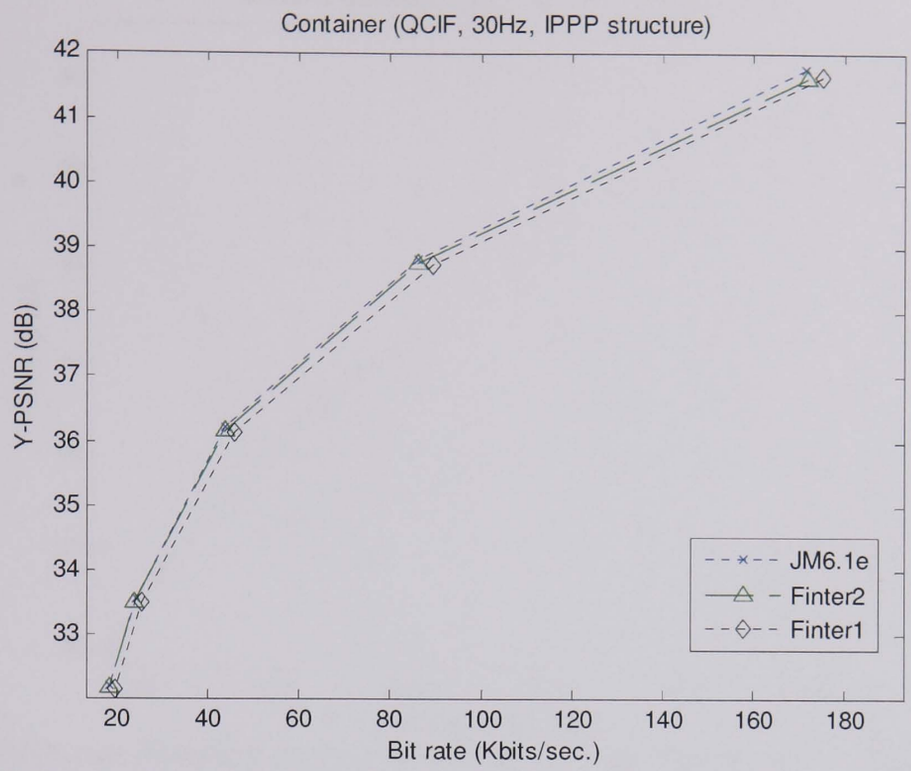


Fig 4-5 PSNR-rate distortion curves for the *Container* (Class A) obtained by the JM6.1e algorithm and the proposed *Finter1* and *Finter2* algorithms.

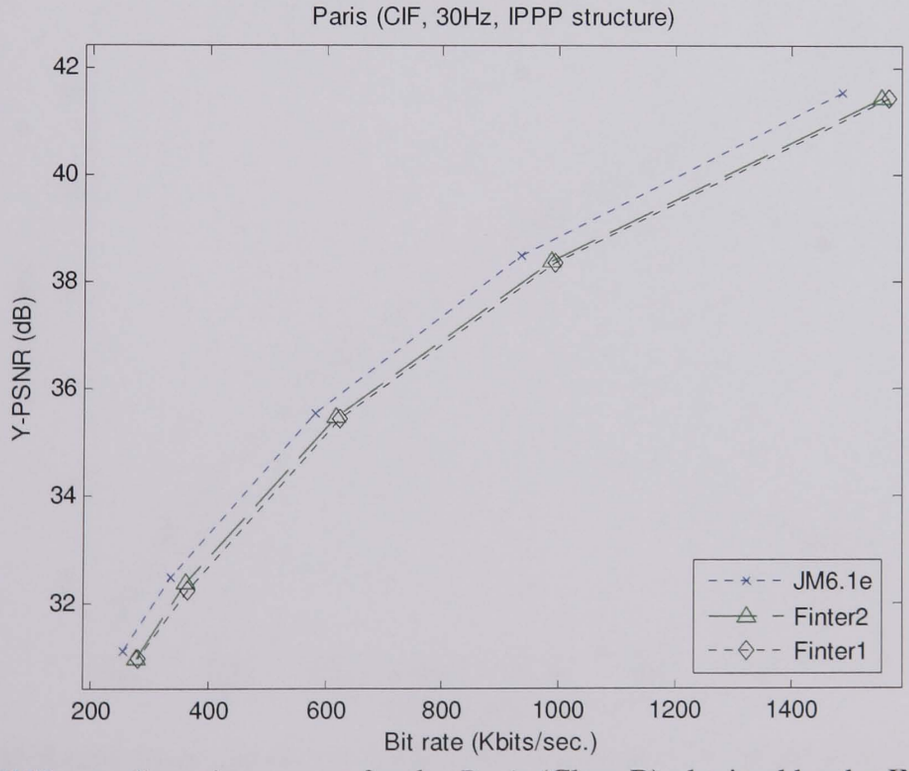


Fig 4-6 PSNR-rate distortion curves for the *Paris* (Class B) obtained by the JM6.1e algorithm and the proposed *Finter1* and *Finter2* algorithms.

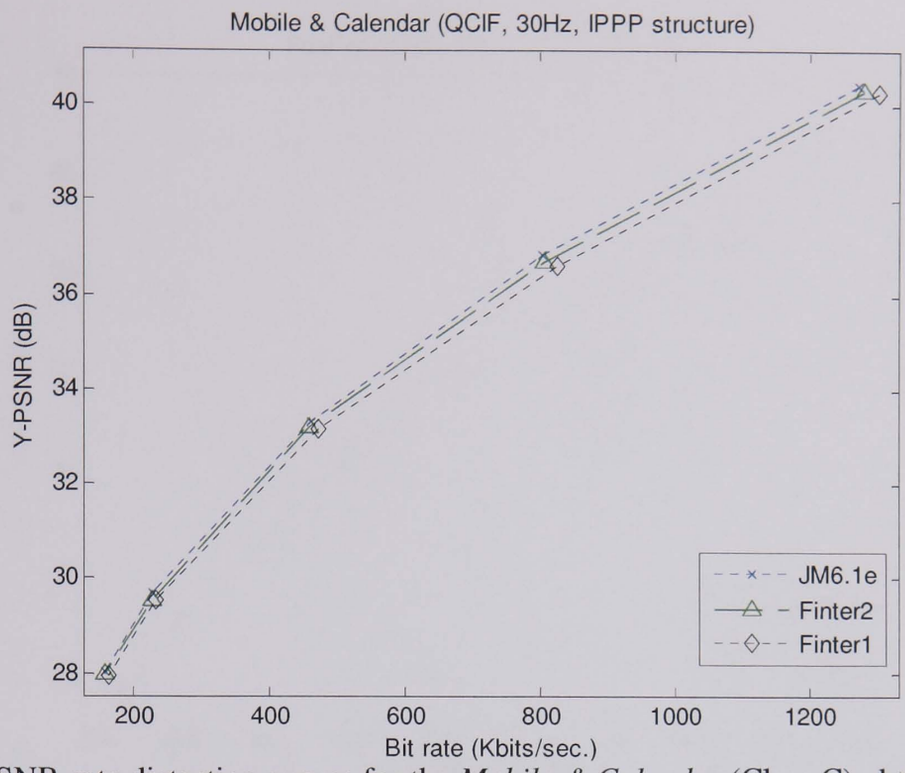


Fig 4-7 PSNR-rate distortion curves for the *Mobile & Calendar* (Class C) obtained by the JM6.1e algorithm and the proposed *Finter1* and *Finter2* algorithms.

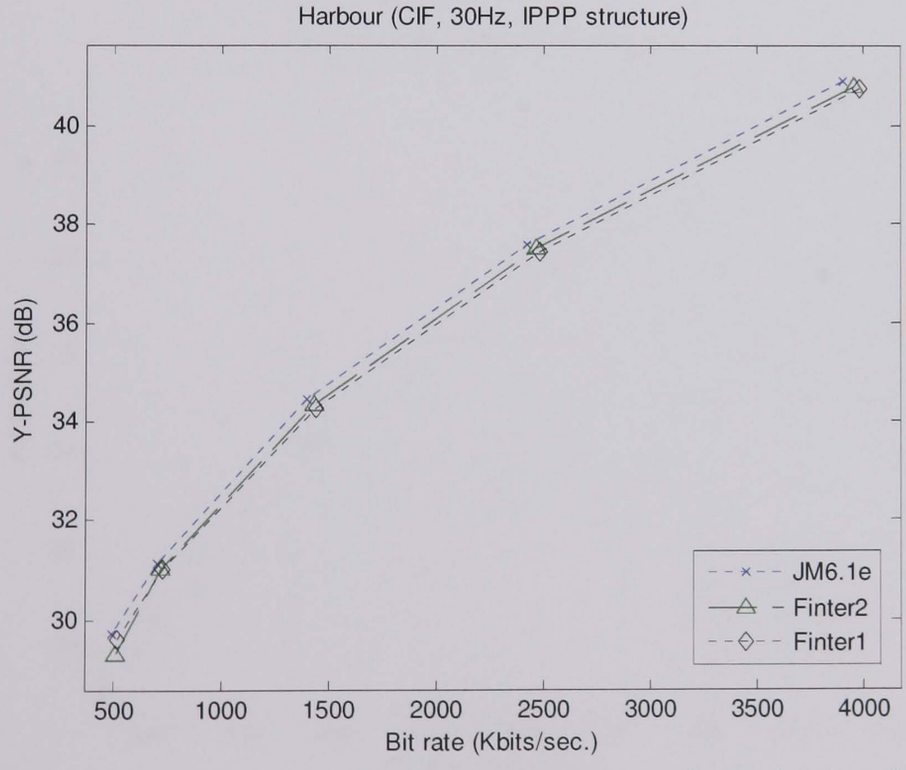


Fig 4-8 PSNR-rate distortion curves for the *Harbour* (Class C) obtained by the JM6.1e algorithm and the proposed *Finter1* and *Finter2* algorithms.

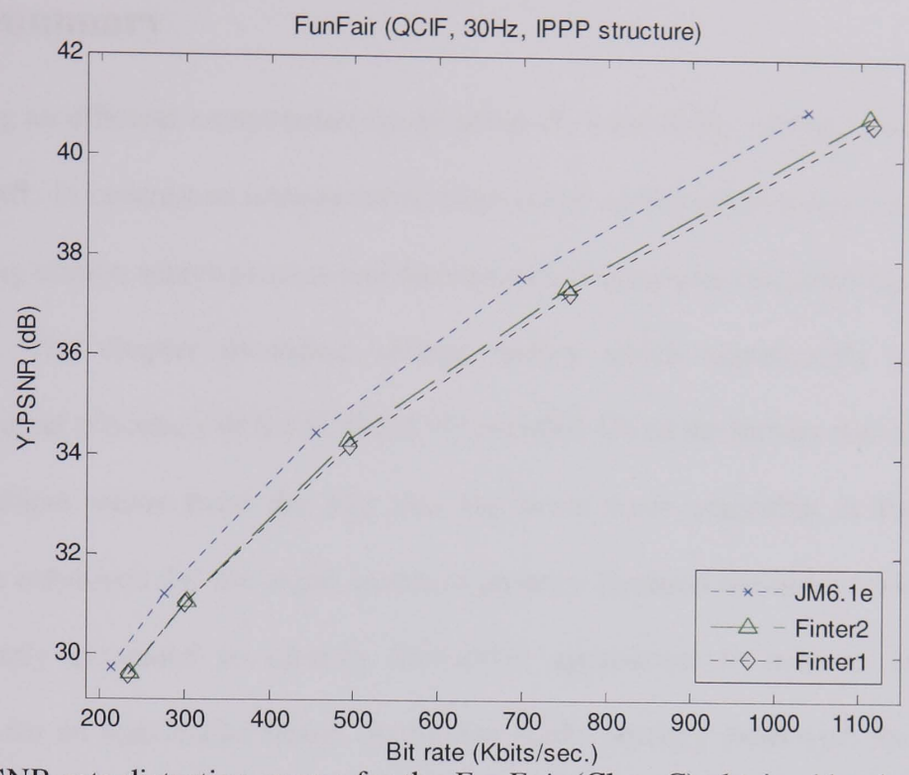


Fig 4-9 PSNR-rate distortion curves for the *FunFair* (Class C) obtained by the JM6.1e algorithm and the proposed *Finter1* and *Finter2* algorithms.

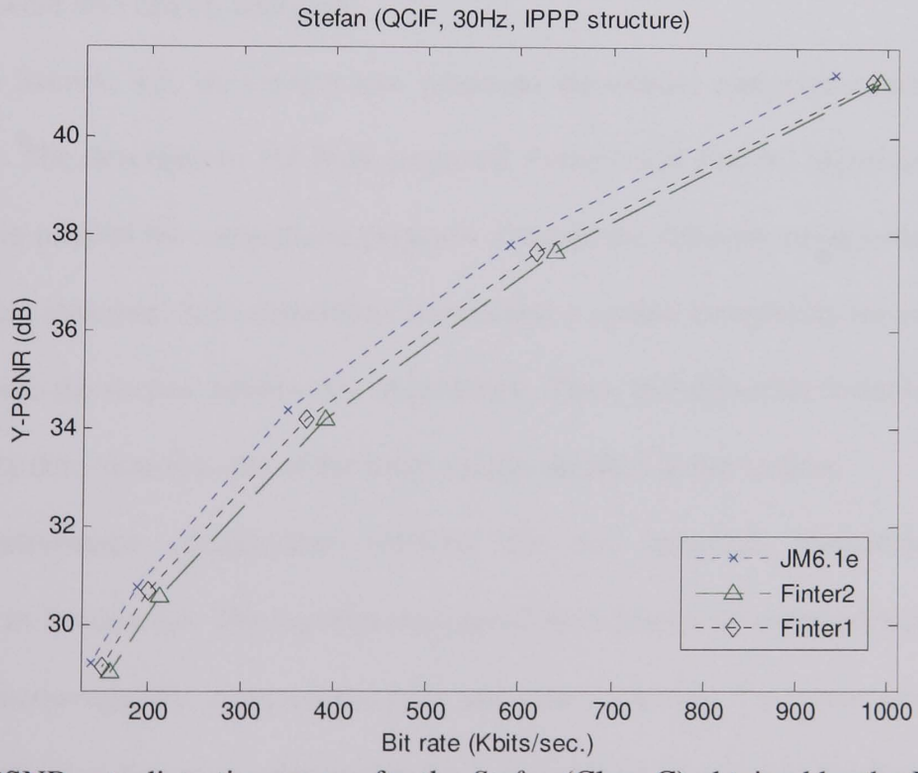


Fig 4-10 PSNR-rate distortion curves for the *Stefan* (Class C) obtained by the JM6.1e algorithm and the proposed *Finter1* and *Finter2* algorithms.

4.4 Summary

Achieving an efficient computation for H.264/AVC inter-frame coding is the topic of this chapter. In contrast to intra-pictures, inter-frame coding necessitates both a time-demanding motion search process and an expensive Lagrangian examination for mode selection. The chapter identified several factors which significantly affect the computational efficiency of the H.264/AVC encoder. Of all the factors considered, the main problem stems from the fact that the brute force algorithm is the current technique employed for the mode selection process. Detailed literature reviews were subsequently presented to identify alternative approaches to achieve fast mode selection for an inter-coded frame. According to the strategy employed, the existing algorithms can be roughly classified into two categories. The pros and cons of each category were also briefly analysed.

In Section 4.3, we present two proposed algorithms comprising hierarchical decisions. The descriptions for both proposed *Finter1* and *Finter2* algorithms were recorded in parallel for comparison purposes. Despite the different composition in the hierarchical structure, both algorithms necessitate a spatial complexity measurement to determine the texture details of a macroblock. Thus, the algorithm formulation for complexity detection was one of the main studies detailed in the section.

Performance comparisons between the two proposed algorithms were presented in Section 4.4. The experiments placed an emphasis on coding efficiency, in terms of picture quality, compression ratio and time reduction. Extensive simulations to obtain the rate-distortion relationship for a wide range of compression levels were performed. The experimental results verify that both algorithms achieve a time reduction in the encoding of inter-frames. This is achieved without introducing significant visual degradations, determined by the objective PSNR measurement.

Chapter 5

Improved Scheme for Multiple Mode Selection

We discussed two algorithms for efficient inter-frame implementation in the previous chapter. The success of the proposed approaches is attributed to the hierarchical structure which discriminates different kinds of encoded macroblocks. Both algorithms were shown to achieve a significant reduction in computation time [83]. However, the performance gains of both algorithms were dependent on the spatial content and amount of motion in the video sequence, for a limited range of compression levels. In this chapter, we propose improved schemes [80, 81] based on the *Finter2* algorithm. The proposed advanced algorithm is described as a hierarchical structure comprising three levels. Each level features a more sophisticated search process and robust predictions to achieve a better PSNR-rate performance and a reduced computational requirement regardless of the video content and the coding condition. This chapter is arranged as follows: Section 5.1 gives a detailed formulation of the proposed algorithm. The results of extensive simulations are summarised in Section 5.2. Finally, Section 5.3 discusses some overall conclusions.

5.1 The Improved Scheme for the *Finter2* Algorithm

The proposed scheme contains a more thorough search process and robust predictions to select an optimal inter-mode for each macroblock. The scheme is described as comprising three levels. Each level targets a different category of modes according to the complexity of the search process. The following subsections introduce the three levels.

5.1.1 Advance Skipped Macroblock Detector

The first level of the proposed algorithm identifies the macroblocks to be encoded with *SKIP*. It functions in a similar manner to the SKIP-detection strategy introduced in Section 4.1.1 Skipped macroblocks are encoded without the utilisation of motion compensation. Consequently, this simple coding method is likely to incur severe quality degradation within the macroblock if an incorrect identification is made. Moreover, the degradation caused by a wrong prediction is likely to drift temporally, such that the overall rate-distortion performance of the sequence suffers. The Joint Video Specification [33] requires that a set of conditions be satisfied for a macroblock to be encoded with the *SKIP* mode if rate-distortion optimisation (RDO) is excluded, as in the baseline encoder. The criteria are as follows:

1) The best motion compensation block size is 16x16.
2) The reference frame is the previous frame in display order.
3) The best motion vector is the predicted motion vector (PMV) regardless of whether it is zero or not.
4) The transform coefficients of the *SKIP*-mode residue block are all quantised to zero.

If RDO is considered, the mode selected is that which produces the least Lagrangian cost. Condition 4 above can be a good indicator for *SKIP* detection, as it implies motion compensation is not needed. However, verification of this condition requires expensive computation of the DCT and quantised coefficients. Thus, several studies [42, 74, 76, 77, 83] have proposed alternative threshold approaches based on a temporal error measurement. The threshold methods achieve fast detection for skipped-macroblocks. However, they are insufficiently robust to adapt to different coding environments, as the decision to adopt skipped-macroblocks is affected by several factors, for instance the choice of *Qp* factor and the outcome of Lagrangian cost evaluation. Due to this effect, a detection strategy incorporating an adaptive threshold and a fast transform-quantised implementation is proposed here. The corresponding block diagram is shown in Fig. 5-1.

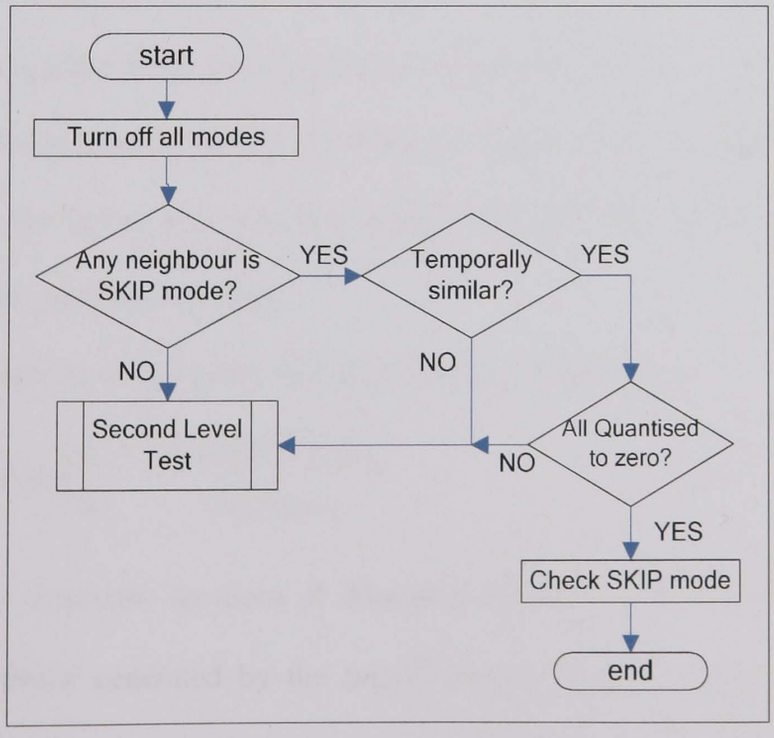
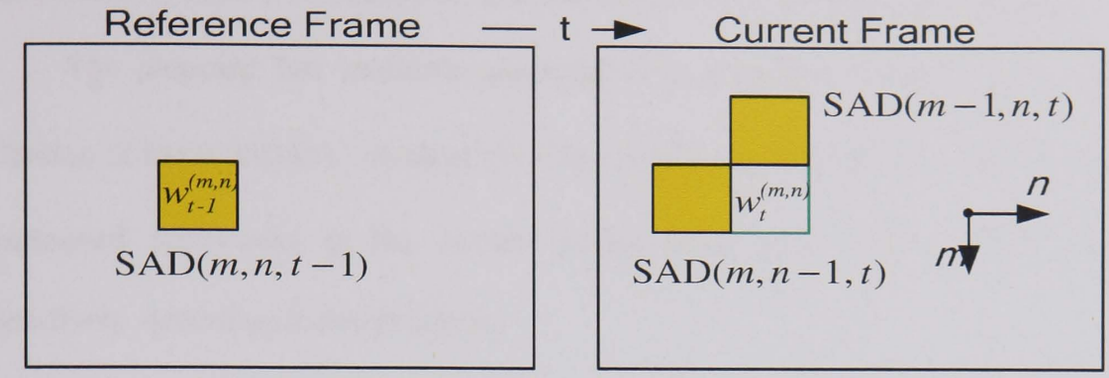


Fig. 5-1 Flowchart of the first level of improved scheme that targets macroblocks encoded by *SKIP* mode.



$$Thd_A = \text{avg}\{\text{SAD}(m,n,t-1) + \text{SAD}(m,n-1,t) + \text{SAD}(m-1,n,t)\}$$

Fig.5-2. Neighbouring macroblocks located in the current frame and the reference frame used for computation of Thd_A.

The purpose of the threshold in the proposed schemes is to identify the residue macroblocks that possess low energy in order to increase the accuracy of the fast transform-quantised evaluation described later. Considering that skipped macroblocks tend to occur in clusters, such as in a patch of static background, the temporal similarity examination is applied if one of the following conditions is valid

1) The collocated macroblock in the reference frame is encoded with *SKIP* mode.
2) At least one of two possible valid skipped macroblocks is found above or to the left of the current macroblock.

The decision process for temporal similarity detection is defined as

$$\text{Decision} = \begin{cases} 1, & \text{SAD}\left(\tilde{\boldsymbol{w}}_t^{(m,n)}\right) \le Thd_A \\ 0, & \text{otherwise} \end{cases} \qquad (5\text{-}1)$$

where SAD(•) represents the Sum of Absolute Difference (SAD); $\tilde{\boldsymbol{w}}_t^{(m,n)}$ reflects the residue macroblock generated by the $(m,n)^{\text{th}}$ current macroblock and its collocated macroblock in the reference frame; Thd_A is an adaptive threshold obtained from the average SAD values of the available skipped neighbours in the current frame and the reference frame. The corresponding locations of the neighbours are illustrated in Fig. 5-

2. A non-zero outcome in (5-1) indicates that the current block is a potential skipped-macroblock. A further examination of its transform-quantised coefficients is required.

The proposed fast transform-quantised implementation is derived from the definition of the H.264/AVC standard [34, 79]. Let $\tilde{W}_q$ and $\tilde{W}_{DCT}$ be the quantised and transformed coefficients of the current residue block, $\tilde{w}_t$ (of size 4x4 pixels), respectively. According to the definition,

$$\tilde{W}_q(i,j)=sign\{\tilde{W}_f(i,j)\}\left\lfloor \frac{\tilde{W}_f(i,j).M(Qp\ \%\ 6,r)}{2^{15+Qp/6}}+f \right\rfloor, \qquad (5\text{-}2)$$

where $0\leq i,j\leq 3$ and $r = 2-(i\ \%2)-(j\ \%2)$

The symbols % and $\lfloor \bullet \rfloor$ are the modular and floor operators, respectively; f represents a dead-zone control parameter which typically lies in the range 0 to ½; and $M(Qp\%6, r)$, the quantisation coefficient for the scaling function, has been pre-defined for each frequency component as a periodic table, M.

$$M = \begin{matrix} r=0 & r=1 & r=2 \end{matrix} \begin{bmatrix} 5243 & 8066 & 13107 \\ 4660 & 7490 & 11916 \\ 4194 & 6554 & 10082 \\ 3647 & 5825 & 9362 \\ 3355 & 5243 & 8192 \\ 2893 & 4559 & 7282 \end{bmatrix} \begin{matrix} \leftarrow (Qp\%6)=0 \\ \leftarrow (Qp\%6)=1 \\ \leftarrow (Qp\%6)=2 \\ \leftarrow (Qp\%6)=3 \\ \leftarrow (Qp\%6)=4 \\ \leftarrow (Qp\%6)=5 \end{matrix}, \qquad (5\text{-}3)$$

The purpose of the detection is to examine whether the coefficients have been quantised to zero. Hence, (5-2) can be rewritten as

$$\frac{\left|\tilde{W}_{DCT}(x,y)\right|.M(Q_M,r)}{2^{15+Q_E}}+f<1 \qquad (5\text{-}4)$$

Table 5-1
Values of $T_{Qz,0}$ and $T_{Qz,1}$ with respect to Qp factor

Qp	20	21	22	23	24	25	26	27
$T_{Qz,0}$	21.7	23.3	26.7	30.0	33.3	36.7	43.3	46.7
$T_{QZ,1}$	33.3	37.5	41.7	47.9	54.2	58.3	66.7	75.0

28	29	30	31	32	33	34
53.3	60.0	66.7	73.3	86.7	93.3	106.7
83.3	95.8	108.3	116.7	133.3	150.0	166.7

where $Q_M \equiv Qp \ \% \ 6$ and $Q_E \equiv Qp \ / \ 6$. f is replaced by 1/6 for inter-frame coding as defined in the H.264 Joint Video Specification [33]. Then, the inequality in (5-4) becomes

$$\left|\tilde{W}_{DCT}(x,y)\right| < \frac{5.2^{14+Q_E}}{3.M(Q_M,r)} \tag{5-5}$$

As the residue blocks have previously passed the threshold examination, they are unlikely to possess high-frequency energy. Thus, the proposed fast implementation is achieved by examining a limited number of low-frequency coefficients, for instance, the first three coefficients in a zig-zag scanned manner, i.e., $\tilde{W}_{DCT}(0,0)$, $\tilde{W}_{DCT}(0,1)$, and $\tilde{W}_{DCT}(1,0)$

$$\text{abs}\left(\begin{bmatrix}\tilde{W}_{DCT}(0,0)\\ \tilde{W}_{DCT}(0,1)\\ \tilde{W}_{DCT}(1,0)\end{bmatrix}\right) < \begin{bmatrix}T_{Qz,0}\\ T_{Qz,1}\\ T_{Qz,1}\end{bmatrix} \tag{5-6}$$

where abs(•) is the absolute operator and T_{QZ} is a transform-quantised threshold fixed for a specific Qp factor. The thresholds can be calculated from (5-6) and stored in advance. Table 5-1 lists the values of $T_{QZ,0}$ and $T_{QZ,1}$ with respect to several Qp values commonly employed in video compression.

The values of the transform coefficients, $\tilde{W}_{DCT}(0,0)$, $\tilde{W}_{DCT}(0,1)$, and $\tilde{W}_{DCT}(1,0)$ can easily be obtained without performing full evaluation of the integer DCT outlined in [79].

$$\tilde{W}_{DCT}(0,0) = \sum_{x=0}^{3}\sum_{y=0}^{3} \tilde{w}_t(x,y) \tag{5-7}$$

$$\tilde{W}_{DCT}(0,1) = \sum_{x=0}^{3}\left(\tilde{w}_t(x,1) - \tilde{w}_t(x,2)\right) + \left\{\sum_{x=0}^{3}\left(\tilde{w}_t(x,0) - \tilde{w}_t(x,3)\right)\right\} << 1 \tag{5-8}$$

$$\tilde{W}_{DCT}(0,1) = \sum_{y=0}^{3}\left(\tilde{w}_t(1,y) - \tilde{w}_t(2,y)\right) + \left\{\sum_{x=0}^{3}\left(\tilde{w}_t(0,y) - \tilde{w}_t(3,y)\right)\right\} << 1 \tag{5-9}$$

where <<1 represents a 1-bit left shift, equivalent to a multiplication by 2. Note that the transform-quantised scheme in the H.264/AVC standard works on the basis of 4x4 pixels. For a macroblock (of size 16x16), the fast transform-quantised evaluation has to be repeated 16 times. Since pixel information is strongly correlated with its neighbours, we propose a skip-and-pick strategy as depicted in Fig.5-3. This reduces the procedure to only four examinations.

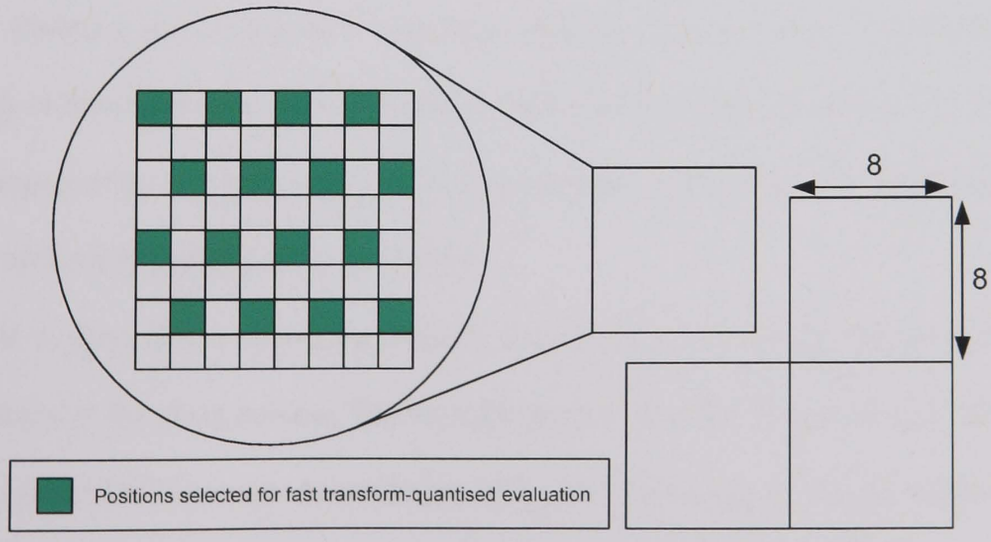


Fig. 5-3 Skip-and-pick strategy to reduce time for examination of transform-quantised coefficients in a macroblock.

The algorithm for this level is summarised as follows:

1.1. Deactivate all inter-modes.

1.2. Determine whether a neighbouring macroblock (as depicted in Fig5-2) is a skipped macroblock. Execute 1.3 if the situation is true. Otherwise, proceed to the next level.

1.3. Perform the temporal similarity test described in (5-1). If a non-zero decision is obtained, continue to examine the current macroblock in 1.4. Otherwise, further examinations in the next level are required.

1.4. Check the inequality (5-6) using (5-7, 5-8, and 5-9) and Table 5-1. If valid, encode the current macroblock with the SKIP mode and record its SAD value. Otherwise, proceed to second level.

5.1.2 Coding for Homogeneous Macroblock

Fig.5-4 shows the flowchart of the second level. This level targets those macroblocks encoded by inter modes with a large partition size (8×16, 16×8, and 16×16 pixels). The general tendency that inter modes with large partition sizes are more suitable for the encoding of homogeneous content has been verified by a number of authors [74, 75, 77, 83]. Consequently, a spatial complexity measurement is developed to determine the content of the macroblocks being considered.

It is clear that a non-homogeneous macroblock features significant intensity dissimilarity in the pixel domain. This is equivalent to the high-frequency (AC) energy reflected in the DCT-domain. According to (4-2), the total energy of the AC coefficients of a macroblock, E_{AC}, can be represented by the variance of the macroblock.

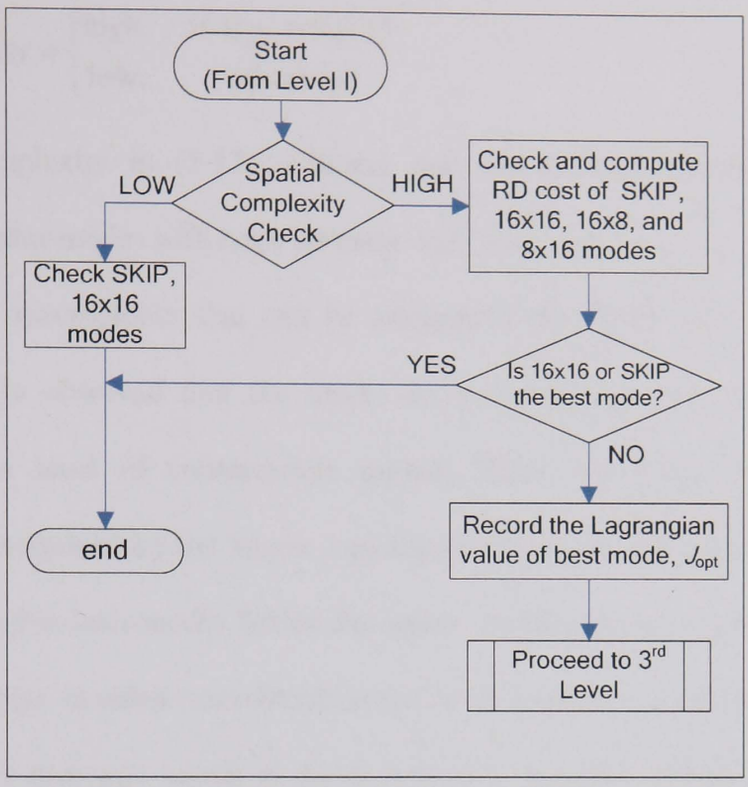


Fig.5-4. Flowchart of the second level that targets macroblocks encoded by inter modes with large partition size.

$$E_{AC}=\sum_{x=0}^{15}\sum_{y=0}^{15}(w_t(x,y))^2-\frac{1}{256}\left(\sum_{x=0}^{15}\sum_{y=0}^{15}w_t(x,y)\right)^2 \tag{5-10}$$

The proposed complexity measurement is made by comparing the variance of the macroblock with the possible maximum variance, $E_{\max}$. The ratio R_{MB} in (4-5) is rewritten as

$$R_{\mathrm{MB}}=\frac{\ln(E_{AC})}{\ln(E_{\max})}\Rightarrow E_{AC}=\exp(R_{MB}\ln(E_{\max}))=E_{\max}^{R_{MB}} \tag{5-11}$$

Note that $E_{\max}$ is obtained from a block comprising a checkerboard pattern in which each adjacent pixel is the maximum and minimum permissible values alternately. By using (4-4), the value of $\ln(E_{\max})$ for a macroblock of size 16x16 is determined to be 15.249.

According to empirical evaluation [83], a macroblock with $R_{\mathrm{MB}} > 0.75$ is considered to be a highly detailed block. Thus, the spatial complexity decision becomes

$$\text{complexity} = \begin{cases} \text{high}, & \text{if } E_{AC} > 92735 \\ \text{low}, & \text{otherwise} \end{cases} \qquad (5\text{-}12)$$

Low spatial complexity in (5-12) indicates that the current macroblock requires examination by inter-modes with large partition size. However, (5-12) excludes the case of high detailed macroblocks that can be adequately described by large partitions. Furthermore, it is observed that the mode decision for these macroblocks varies according to the level of compression applied. Since the mode decision for a macroblock is determined by the lowest Lagrangian (RD) cost, we suggest computing the RD cost of a few inter-modes before the entire search process is performed. If the best mode for a high-detailed macroblock in this level requires a partition size of 16x8 and 8x16, a more thorough search in the next level is required. Otherwise, the mode decision for the current macroblock is made. The detailed algorithm at this level is described as follows:

2.1. *Determine the spatial complexity of the macroblock using (5-10) and (5-12). If the content of the current macroblock is homogeneous, select the best mode from either SKIP or the inter-mode with partition size 16×16. Otherwise, continue to 2.2.*

2.2. *Activate the inter-modes with partition size 8×16, 16×8, and 16×16 and SKIP. Check the current macroblock with the activated inter-modes.*

2.3. *Compute the RD cost of the four modes and obtain a mode decision for this level. If the optimal mode is not SKIP or 16×16, proceed to the third level. Otherwise, the mode decision for the current macroblock is decided.*

5.1.3 Mode Selection for High-detailed Macroblock

The third level computes searches within the *P8x8* mode (including inter-modes with the smaller partition sizes of 8×8, 8×4, 4×8, and 4×4 pixels). Since each decomposed 8-by-8 block has to undergo search operations, a large computational overhead is expected. Fig. 5-5 illustrates the proposed scheme for this level. The current macroblock is decomposed into four non-overlapping 8×8 blocks. Each block is then checked by the inter mode with partition size 8×8. The other inter-modes with smaller partition sizes are activated if the following condition is true

$$J_{8\times8}(N^{th}) < J_{opt}/4 \quad (=J_{TH}) \tag{5-13}$$

where $J_{8x8}(N^{th})$ is the RD cost of the N^{th} current 8×8 block and J_{opt} represents the RD cost of the best mode obtained from the last level. Note that (5-13) does not need to be revised if all inter-modes are activated. Thus, a computational saving is achieved for those first few blocks that do not satisfy the condition in (5-13).

The detailed algorithm at this level is described as follows:

3.1. Decompose the current macroblock into four non-overlapping 8×8 blocks.

3.2. Activate the inter mode with partition size 8×8 only.

3.3. Compute the RD cost for the activated inter-modes for the 8×8 block.

3.4. If all inter-modes have been activated then repeat 3.3 for each remaining 8×8 block, otherwise examine condition (5-13).

3.5. If condition (5-13) is satisfied, activate all the available inter-modes in this level. Repeat 3.3 for each remaining 8×8 block.

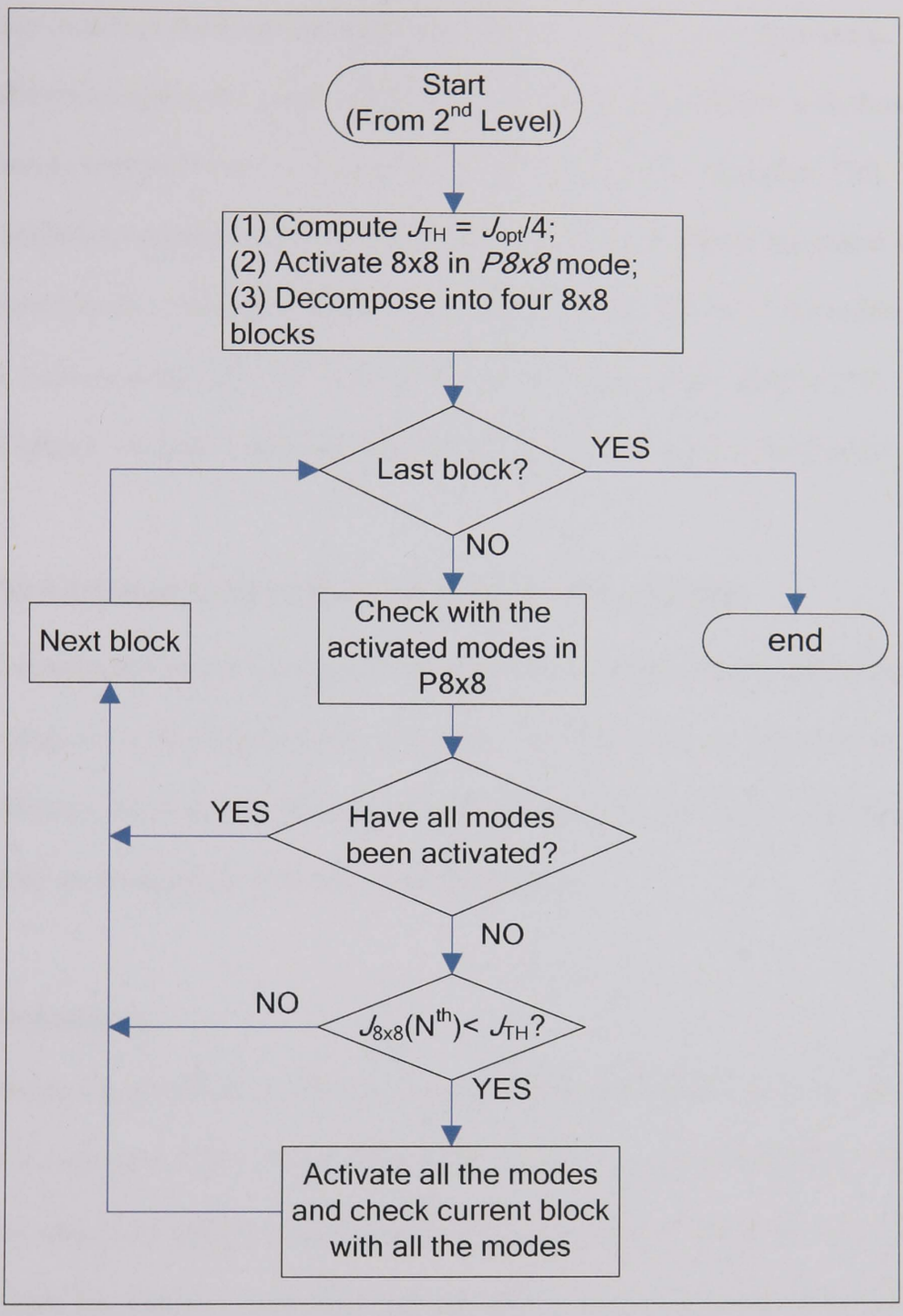


Fig. 5-5 Flowchart of the third level that considers macroblocks encoded with *P8x8* mode.

5.2 Simulations, Comparisons, and Discussions

This section examines the proposed improved scheme [81, 82] in two experiments. The first simulation compares the results of the proposed improved algorithm with those of the previously reported *Finter2* algorithm [78] and Wu et al's algorithm [76]. The second simulation examines the performance of the *IFinter* algorithm integrated with proposed intra-mode selection algorithm, *Fintra* (described in Chapter 3). All results are shown as improvements over the standard H.264/AVC benchmark, JM6.1e [15]. The empirical settings for both simulations are identical to those recorded in Section 4.3.

5.2.1 Performance Comparison for Inter-Frame Coding

Table 5-2 summarises the performance of the algorithms in comparison to JM6.1e inter-frame coding, for a fixed quantisation level, *Qp* =32. Comparisons are given for Y-PSNR difference (measured in dB) and bit rate difference (expressed as a percentage). Table entries are arranged according to class of sequence.

Picture Degradation

General trends are identified as follows. Similar PSNR performance and bit rates are achieved in each case. Class A and Class B test sequences exhibit marginal Y-PSNR differences which are below 0.08dB and 0.11dB, respectively. Wu et al's algorithm performs best for Class C sequences but the gain over the proposed algorithm is insignificant at 0.08dB maximum. A slightly higher picture degradation of between 0.06dB and 0.19dB is observed for the *Finter2* algorithm when coding Class C sequences.

Compression Performance

Considering compression ratio, the bit rate differences for both Wu et al's algorithm and the proposed algorithm are within an acceptable range of less than 4%. In contrast, the performance of the *Finter2* algorithm shows a wider variation depending on the test sequence. The bit rate difference is especially high for some Class C sequences, for example, *Stefan* and *Train & Tunnel*. The algorithm integrating the improved schemes has significantly increased the performance of the *Finter2* algorithm in terms of Y-PSNR and compression ratio. Moreover, the performance is approximately the same as that of Wu et al's method, considered to be one of the best performing algorithms in the literature.

PSNR-rate Relationship Diagrams

The performance of the three fast algorithms is compared with the JM6.1e encoder for a wide range of compression levels. Fig.5-6 to Fig.5-11 show the PSNR-rate relationship diagrams for six test sequences selected from different classes. The curve with the 'X' marker represents the performance of the JM6.1e benchmark. It also reflects an upper bound for the other three algorithms.

In general, the PSNR-rate performance of both Wu et al's algorithm and the proposed method are virtually identical to that provided by the JM6.1e software. This is true at all bit rates. In contrast, the previously reported *Finter2* algorithm shows a decreased PSNR performance at the higher bit rates. The performance degradation is especially noticeable in some Class C's sequences, for example *Fun Fair* and *Stefan*. We therefore conclude that the new algorithm provides a better performance than the *Finter2* algorithm at all bit rates.

Table 5-2
Simulation results for three fast inter-mode selection algorithms, each compared with the H.264 JM6.1e software. (Except were stated, the sequences are of QCIF resolution)

Sequences	Y-PSNR Gain (dB)			Bit Rate Difference (%)		
	Finter2	Wu et al's	Proposed	Finter2	Wu et al's	Proposed
Class A						
Akiyo	-0.08 dB	-0.03 dB	**-0.04 dB**	-0.96%	-1.16%	**-1.77%**
Container	-0.04 dB	-0.01 dB	**-0.03 dB**	-0.94%	0.04%	**-1.44%**
Hall Monitor	-0.02 dB	-0.01 dB	**-0.03 dB**	6.02%	2.68%	**2.04%**
Class B						
Paris (CIF)	-0.11 dB	-0.06 dB	**-0.05 dB**	7.19%	1.96%	**2.83%**
Silent Voice	-0.07 dB	-0.07 dB	**-0.03 dB**	5.02%	1.12%	**0.24%**
News	-0.11 dB	-0.04 dB	**-0.06 dB**	3.77%	1.94%	**1.99%**
Class C						
Fun Fair	-0.13 dB	-0.05 dB	**-0.12 dB**	9.48%	1.12%	**2.73%**
Harbour (CIF)	-0.12 dB	-0.01 dB	**-0.09 dB**	2.86%	0.96%	**0.76%**
Mobile & Calend	-0.14 dB	-0.04 dB	**-0.11 dB**	0.57%	0.09%	**-0.28%**
Stefan	-0.19 dB	-0.05 dB	**-0.06 dB**	12.98%	0.72%	**1.07%**
Tempete	-0.11 dB	-0.06 dB	**-0.10 dB**	0.12%	1.37%	**0.35%**
Train & Tunnel	-0.06 dB	-0.04 dB	**-0.03 dB**	18.10%	4.12%	**3.59%**
Waterfall (CIF)	-0.06 dB	-0.01 dB	**+0.02 dB**	3.88%	2.59%	**2.78%**
Averages	**-0.09 dB**	**-0.03 dB**	**-0.06 dB**	**5.24%**	**1.35%**	**1.15%**

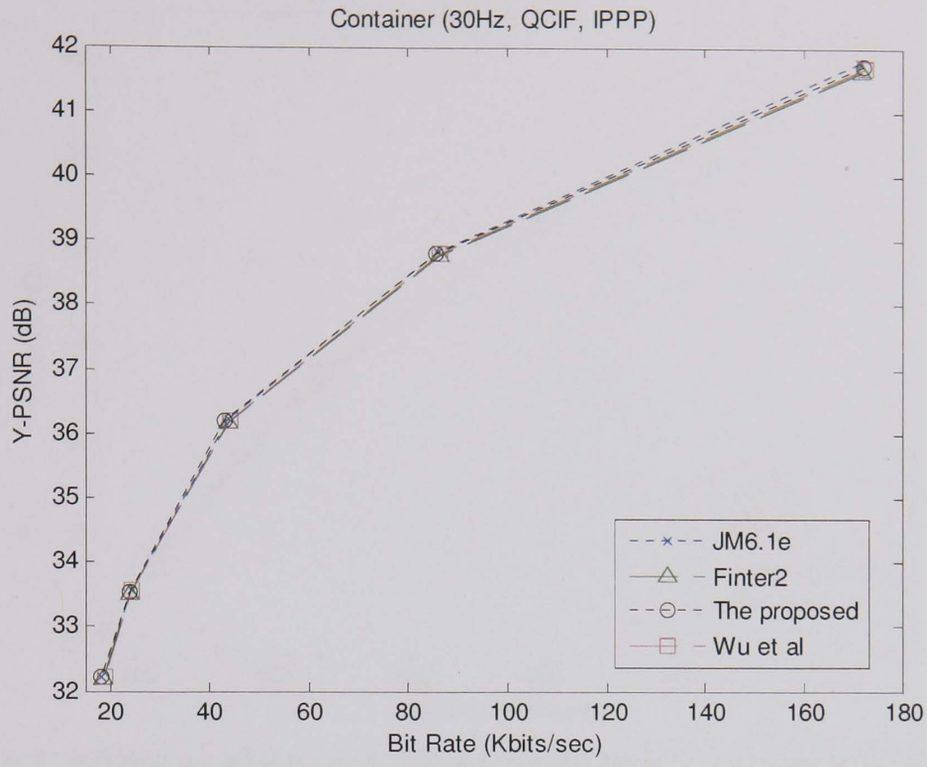


Fig 5-6 PSNR-rate relationship diagram for the *Container* (Class A) sequence.

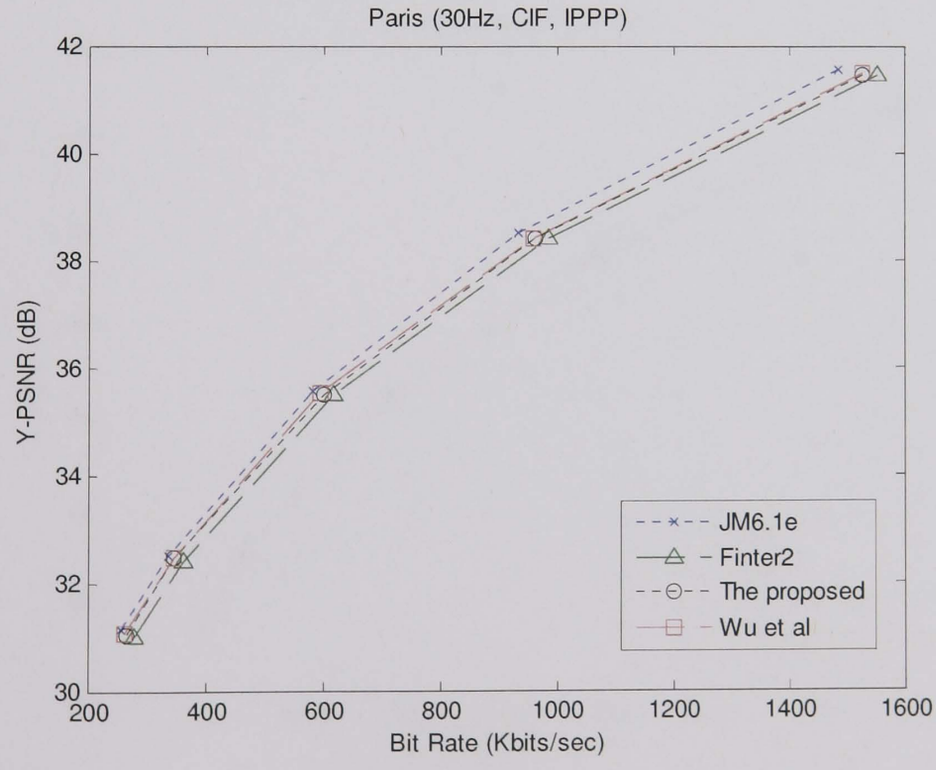


Fig 5-7 PSNR-rate relationship diagram for the *Paris* (Class B) sequence.

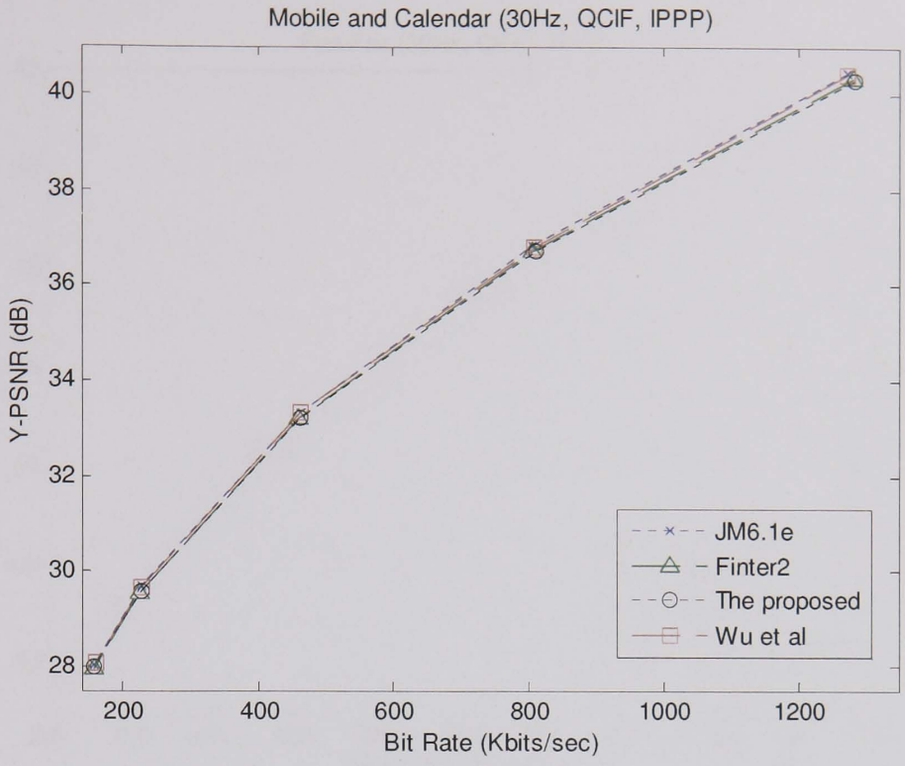


Fig 5-8 PSNR-rate relationship diagram for the *Mobile and Calendar* (Class C) sequence.

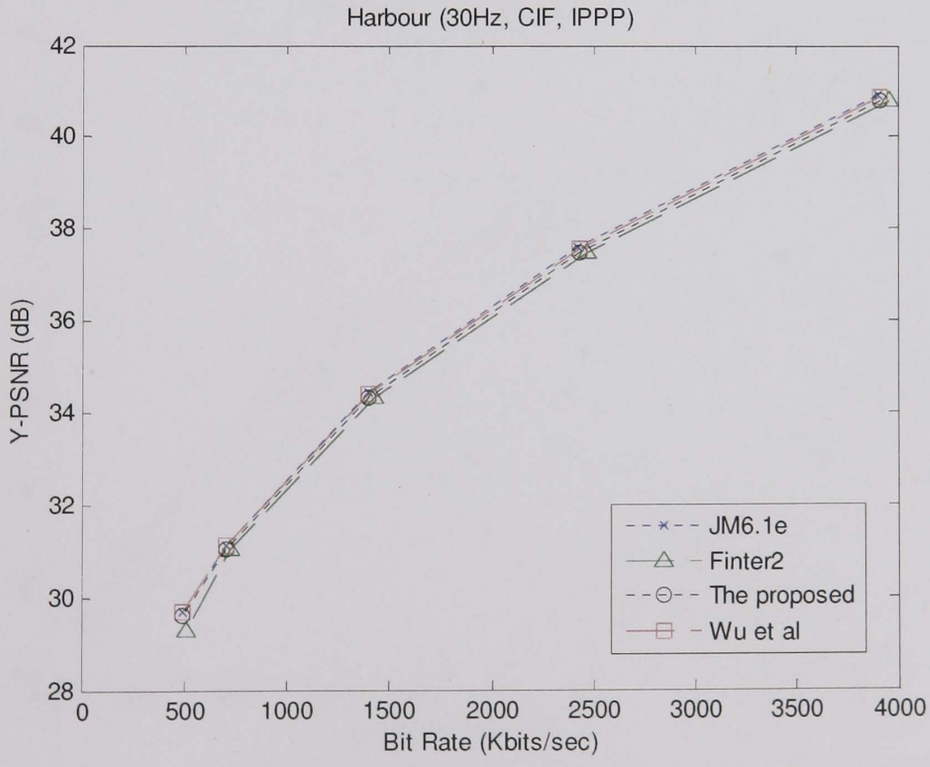


Fig 5-9 PSNR-rate relationship diagram for the *Harbour* (Class C) sequence.

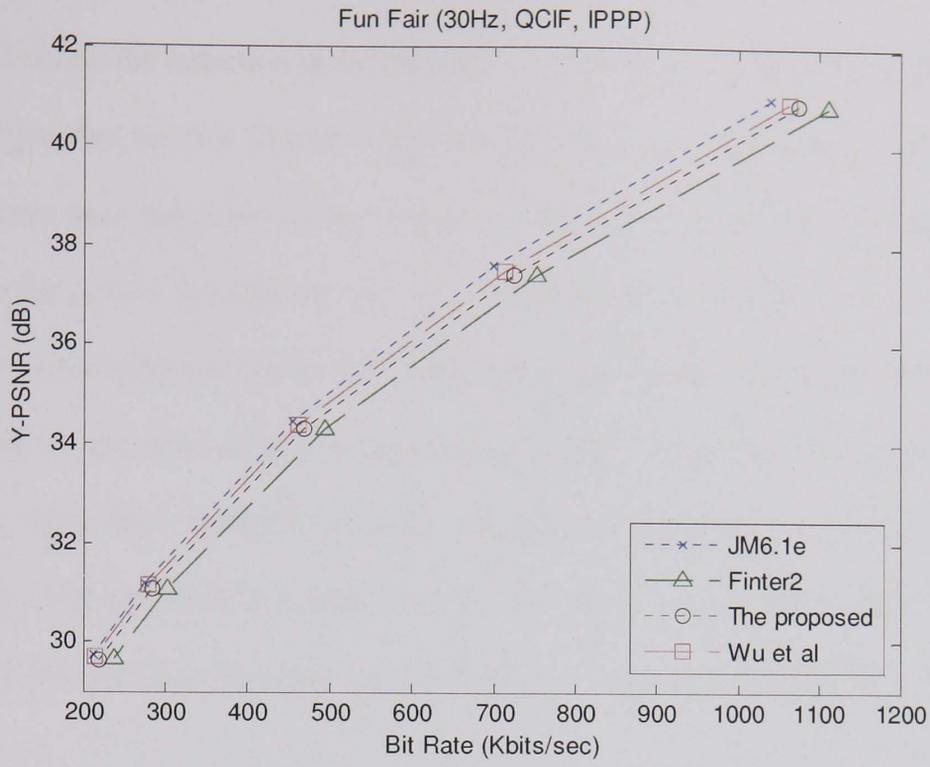


Fig 5-10 PSNR-rate relationship diagram for the *Fun Fair* (Class C) sequence.

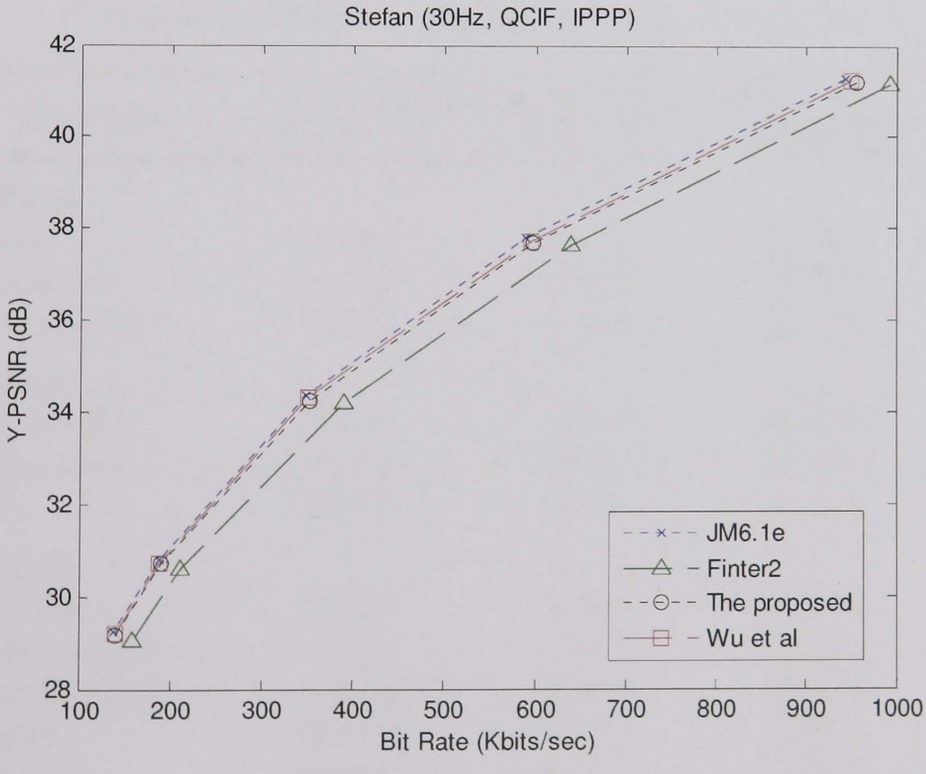


Fig 5-11 PSNR-rate relationship diagram for the *Stefan* (Class C) sequence.

Computational Saving

Table 5-3 shows the reduction in coding time for the three fast algorithms, as defined in (2-29). Note that the fast motion estimation algorithm recommended for the JM6.1e software has been integrated in the simulation. This explains why the speedup figures given for the *Finter2* algorithm in [50] are different to those in Table 5-3.

It is clear that each of the fast algorithms attain different degrees of time saving depending on the class of test sequence. In general, an increased saving is usually obtained when little motion is present, whereas the sequences in Class C pose a challenge. The proposed algorithm provides the best computational saving. This is especially true for Class C sequences where the new scheme performs twice as well as

Table 5-3
Computational reduction compared to the JM6.1e encoder.

Sequences	Computational Saving (ΔT) cf. JM6.1e (%)		
	Finter2 [7]	Wu et al's	**Proposed**
Class A			
Akiyo	48.05%	47.53%	**72.70%**
Container	43.00%	38.78%	**72.91%**
Hall Monitor	47.04%	34.24%	**75.27%**
Class B			
Paris (CIF)	35.63%	26.99%	**67.96%**
Silent Voice	36.23%	25.04%	**68.75%**
News	39.94%	24.03%	**67.94%**
Class C			
Fun Fair	13.80%	4.03%	**40.54%**
Harbour (CIF)	17.01%	15.95%	**61.30%**
Mobile & Cal.	15.05%	6.55%	**43.72%**
Stefan	24.41%	24.07%	**55.25%**
Tempete	19.86%	17.11%	**54.52%**
Train & Tunnel	23.58%	23.74%	**60.32%**
Waterfall (CIF)	34.32%	25.32%	**67.85%**
Averages	**30.61%**	**24.11%**	**62.23%**

the other two algorithms. Computational savings of 40% - 68% are obtained, compared with 13% - 34% for *Finter2* and 4% - 25% for Wu et al's algorithm.

Subjective Examination

Fig.5-12 shows snapshots taken from the *Stefan* sequence, for the purpose of subjective examination. The four snapshots are generated by the JM6.1e codec (34.69dB), the *Finter2* algorithm (34.48dB), Wu et al's algorithm (34.63dB), and the proposed scheme (34.53dB), respectively. To show more detail, the magnified part is focused at the upper body of the tennis player. There is no evidence of visually perceptible degradation even in objects undergoing significant movement which results in a Y-PSNR distortion of no more than 0.21dB.

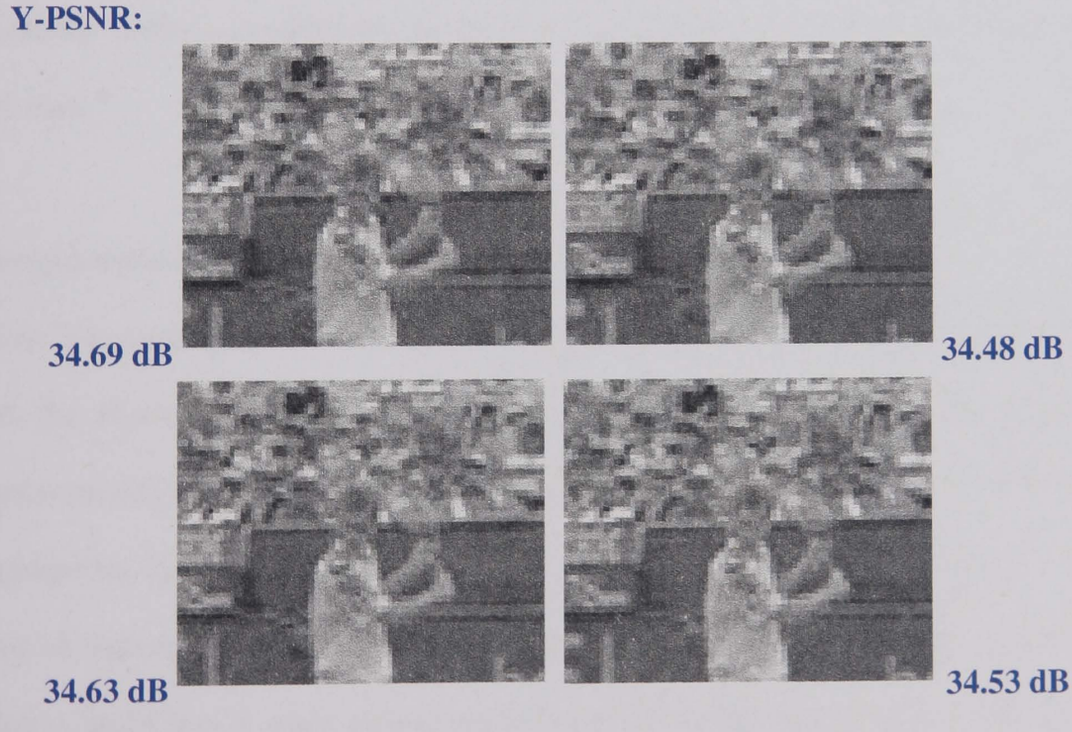


Fig.5-12. Magnified snapshots from frame #9 of the *Stefan* sequence with *Qp*=28. (Top) encoded by JM6.1e; the *Finter2* method; (Bottom) Wu et al's algorithm; the proposed scheme.

5.2.2 Performance Comparison for Hybrid Coding Structure

We further examine the performance of the proposed algorithm incorporating the *Fintra* algorithm described in Chapter 3. Note that the empirical settings for the simulation are identical to those recorded in Section 4.3

PSNR-rate Relationship Diagrams

The six figures, from Fig. 5-13 to Fig. 5-18 show the rate-distortion performance of the three encoders, i.e., JM6.1e benchmark, the proposed improved *Finter* algorithm (denoted as *IFinter*), and the combination of both *IFinter* and *Fintra* algorithms. The combined algorithm provides no evidence of degradation in any objective measurements when compared to the *IFinter* method, as the performance of both curves are absolutely identical at all bit rates. In short, the integration of *Fintra* algorithm forms a complement to the *IFinter* algorithm for coding sequences from any class.

Computational Speedup

Table 5-4 compares the computational performance between the combined algorithm and the *IFinter* encoder in terms of time reduction. The results are shown as improvements over the JM6.1e codec. It is noted that further computational savings acquired by the combined algorithm over the *IFinter* encoder are dependent on the class of sequence. A saving difference of at most 7% is reported for the sequences in Class A and Class B when employing the combined algorithm instead of the *IFinter* method. In contrast, a more significant saving of up to 15% is provided when encoding Class C sequences. The explanation for this unequal saving ratio is straightforward: the *Fintra* algorithm shows more benefit to the Class C sequences

where the intra-coded blocks are prevalent. In conclusion, the integrated encoder featuring the *Fintra* and *IFinter* algorithms successfully exploit further complexity reduction, such that the time requirement for coding any sequence is at least 50% of that needed by the JM6.1e encoder.

Table 5-4
Computational reduction of combined algorithm compared to the JM6.1e encoder.

Sequences	Computational Saving (ΔT) cf. JM6.1e (%)		
	IFinter	Combined Alg.	% Difference
Class A			
Akiyo	72.70%	74.00%	1.30%
Container	72.91%	75.85%	2.94%
Hall Monitor	75.27%	77.69%	2.42%
Class B			
Paris (CIF)	67.96%	73.56%	5.60%
Silent Voice	68.75%	70.86%	2.11%
News	67.94%	74.66%	6.72%
Class C			
Fun Fair	40.54%	53.75%	13.21%
Harbour (CIF)	61.30%	71.30%	10.00%
Mobile & Cal.	43.72%	59.08%	15.36%
Stefan	55.25%	65.34%	10.09%
Tempete	54.52%	65.51%	10.99%
Train & Tunnel	60.32%	67.64%	7.32%
Waterfall (CIF)	67.85%	71.92%	4.07%
Averages	**62.23%**	**69.32%**	**7.09%**

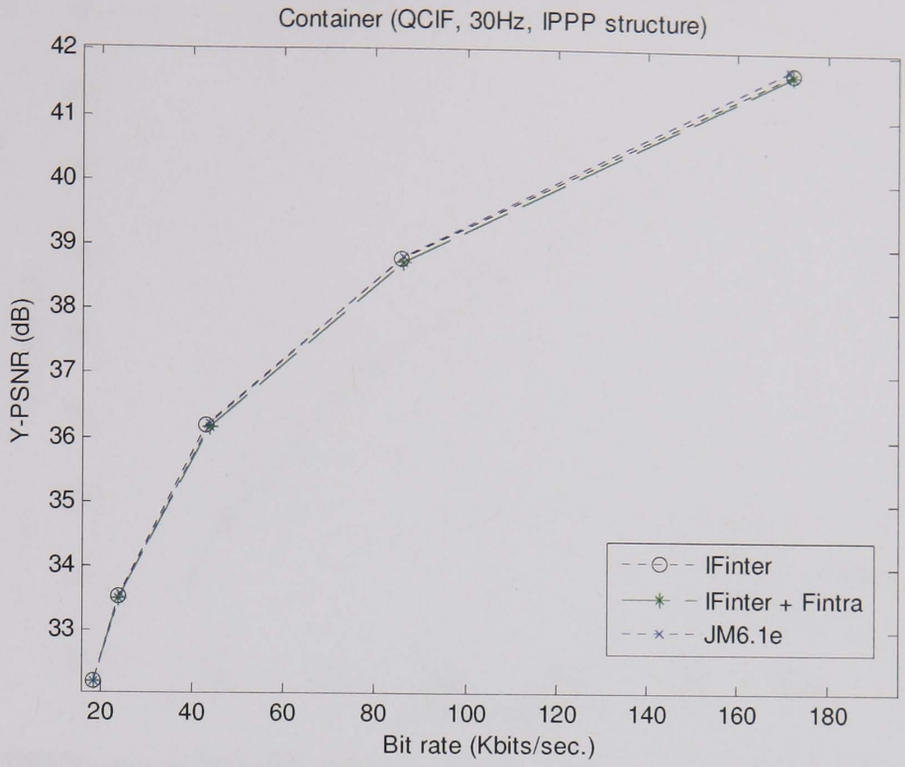


Fig 5-13 PSNR-rate relationship diagram for the *Container* (Class A) sequence.

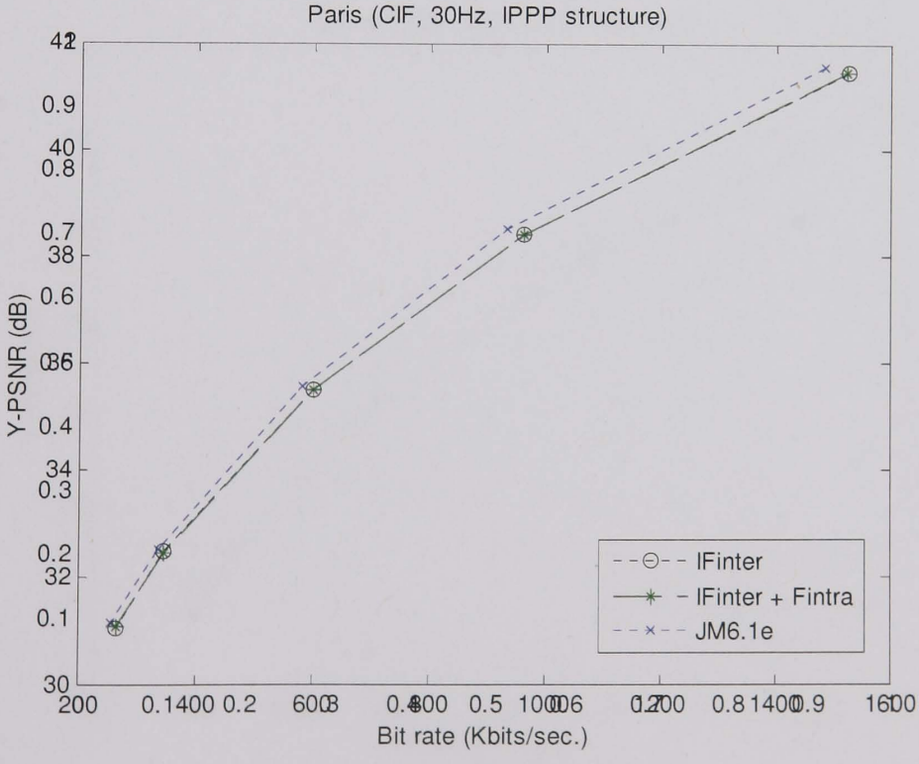


Fig 5-14 PSNR-rate relationship diagram for the *Paris* (Class B) sequence.

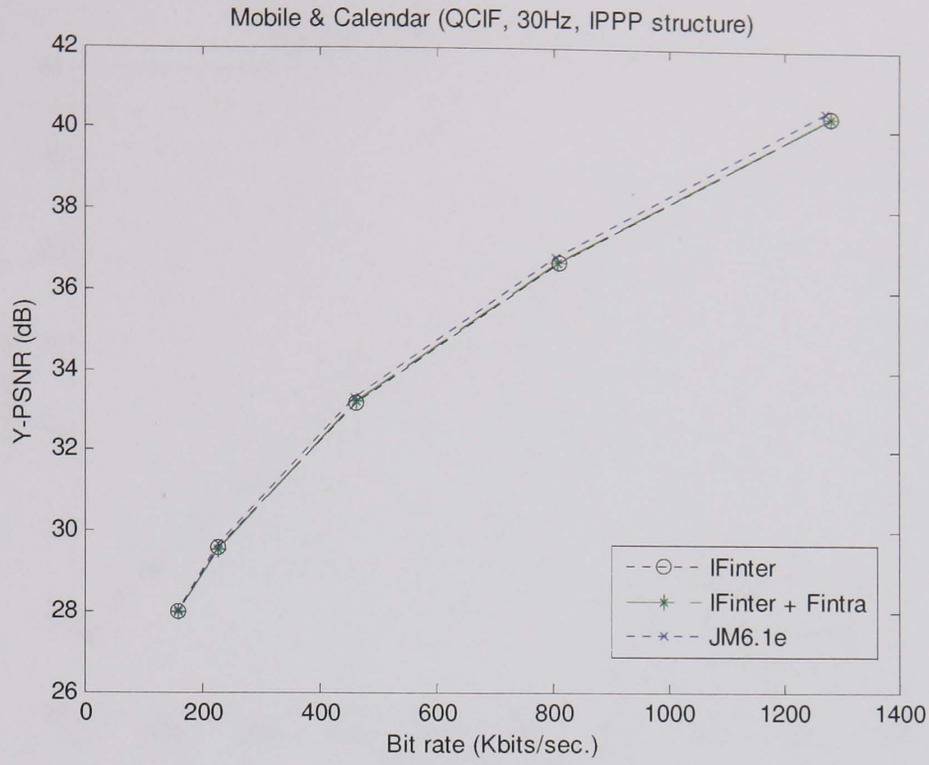


Fig 5-15 PSNR-rate relationship diagram for the *Mobile and Calendar* (Class C) sequence.

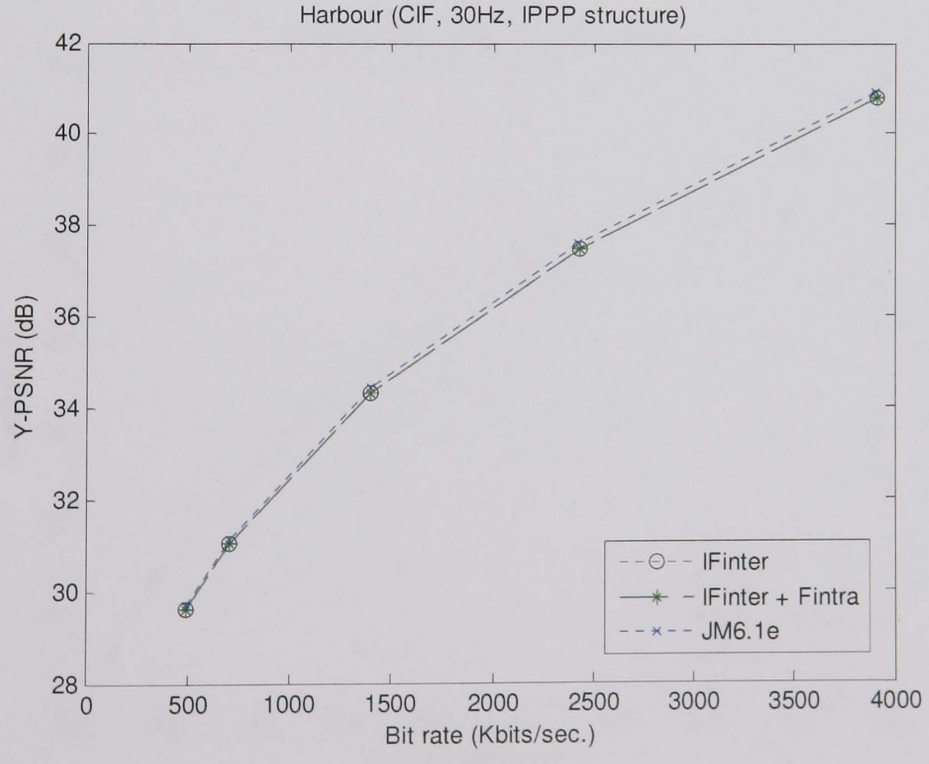


Fig 5-16 PSNR-rate relationship diagram for the *Harbour* (Class C) sequence.

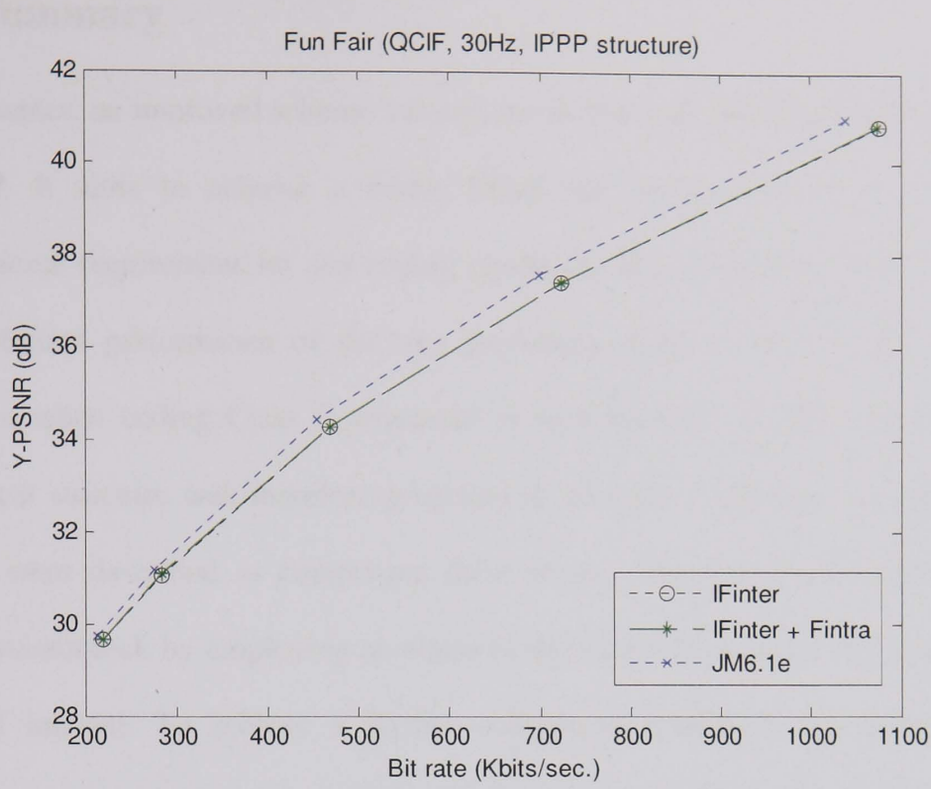


Fig 5-17 PSNR-rate relationship diagram for the *Fun Fair* (Class C) sequence.

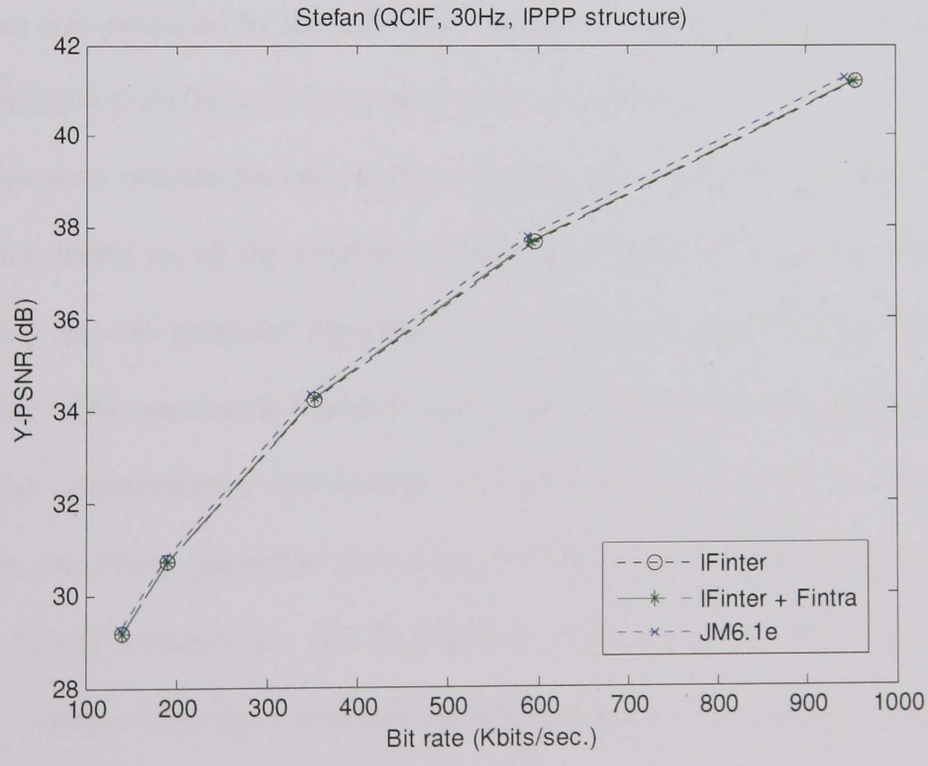


Fig 5-18 PSNR-rate relationship diagram for the *Stefan* (Class C) sequence.

5.3 Summary

In this chapter, an improved scheme tailored for H.264/AVC inter-frame coding was described. It aims to achieve a better PSNR-rate performance and a reduced computational requirement for any coding condition. The chapter initially identified the insufficient performance of the two previously reported *Finter1* and *Finter2* algorithms when coding Class C sequences at high bit-rates. A more sophisticated hierarchical structure was therefore proposed to provide a solution. The proposed schemes were described as comprising three levels. The first level identifies the skipped macroblock by employing an adaptive threshold check and a fast transform-quantised method. To achieve a further complexity reduction, the second level discriminates high-detailed macroblocks which can be adequately described by large partitions. Lastly, the motion search within the *P8x8* mode depends on the outcome of Lagrangian cost produced by the individual subblock, such that the inter-modes with small partition size can be selectively performed accordingly.

Extensive simulations have been performed in two ways with a wide range of compression levels for all the sequence classes. The results from the first simulation demonstrate that the proposed algorithm has significantly improved the PSNR-rate performance of the previously reported *Finter2* algorithm at all bit rates, while almost halving the computational requirement for all sequence classes. In the second simulation, the *Fintra* algorithm for coding intra-blocks has been integrated into a complete hybrid structure for fast H.264/AVC implementation. The experimental outcomes confirm that the combined algorithm achieves a further computational reduction of up to 15% without sacrificing the degradation in any objective measurement. More importantly, the combined algorithm ensures a reduction in

computation time of at least 50% compared with the JM6.1e software. This is achieved regardless of the sequence class and any pre-defined coding conditions.

Chapter 6

Mesh-based Motion Compensation Featuring a Dual Layer Structure

6.1 Overview

The development of video coding at low bit rates is a key technical issue in the implementation of new applications such as mobile multimedia and video streaming over low bandwidth channels. The MPEG-x and H.26x video coding standards adopt a conventional hybrid coding approach, employing block matching (BMA) motion compensation [46] and the discrete cosine transform (DCT) [23]. The reasons are that (a) a significant proportion of the motion trajectories found in natural video can be approximately described with a rigid translational motion model; (b) fewer bits are required to describe simple translational motion; (c) implementation is relatively straightforward and amenable to hardware solutions. However, in spite of its relatively low computational requirement, the block-matching algorithm can provide visually unacceptable results when motion trajectories exhibit non-translational motion. The difficulties stem from the fact that connectivity of the vertices of adjacent blocks is not preserved [88].

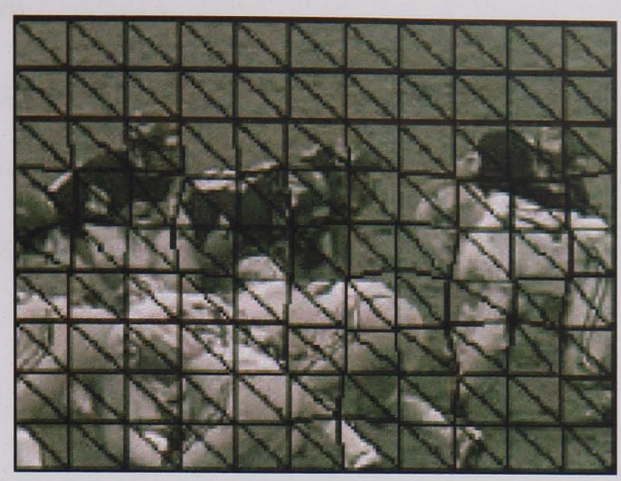


Fig. 6-1 *Football* sequence predicted by deformed triangular mesh of size 16x16.

In contrast, the use of mesh-based motion prediction provides an alternative when coping with non-rigid translational motion. The procedure is to divide the current frame into a number of triangular patches [89] or rectangular patches [90], and to find the best matching corresponding patch in the reference frame when deformed by affine transformation, as shown in Fig. 6-1. The mean absolute difference is commonly used as the matching criterion, and the motion is defined by the change in position of the corresponding vertices [85]. Compared with a disjointed collection of blocks in BMA, mesh-based motion models maintain the connectivity of the patches and provide more visually acceptable results. Furthermore, motion is estimated more accurately due to the utilisation of an affine transformation, which supports various spatial deformations, such as translation, rotation, and zoom [86]. The deformed patches are described by the motion displacement of the grid points (the points shared by the vertices of the patches) for motion coding purposes.

Several studies have been done to provide solutions to compute the motion vector for a grid point. Basically, they can be classified as search-based [84, 91, 92, 93], using an optimisation framework [94, 101, 102] and employing a close-form solution [95, 96]. In particular, Nakaya and Harashima [84] proposed a hexagonal matching

algorithm (HMA) using iterative local minimisation for the computation of the prediction error. Wang et al. [94] resolved the problem with the utilisation of adaptive mesh models incorporating energy optimisation. Szeliski et al. [95] suggested a global close form solution with the utilisation of conjugate gradient optimisation.

6.1.1 Mesh size v. Rate-distortion Performance

Intuitively, a better rate-distortion performance is expected when finer meshes are utilised. Fig. 6-2 provides evidence of this by showing the coding results for the *Football* sequence when triangular patches of size 32x32, 16x16, and 8x8 pixels are employed. It is observed that the utilisation of a 16x16 mesh achieves a better rate-distortion performance than using a 32x32 mesh. However, a decrease in mesh size does not necessarily lead to a better identification of motion-active regions. This is due to the connectivity constraint of the fine mesh that reduces the effective area for the motion search. Fig. 6-2 further demonstrates the decrease in PSNR achieved by the employment of 8x8 meshes at low bit rates. This is attributed to the overhead of an increased number of motion vectors which is certainly a disadvantage with regard to compression. To reduce the overhead for motion estimation, Huang and Hsu [97] proposed a hierarchical structure based on quad-tree decomposition, Van Beek et al. [92] employed content-based meshes based on Delaunay mesh simplification, and Cui et al [98] performed motion compensation in the redundant wavelet domain. In a previous study [87], we proposed an algorithm incorporating partial refinement by the employment of two mesh topologies. The technique is based on the utilisation of meshes featuring multiple scales to cope with different prediction demands in the picture. Improved schemes are integrated in a complete codec to improve the rate-distortion performance when the video sequence features widely different amounts of

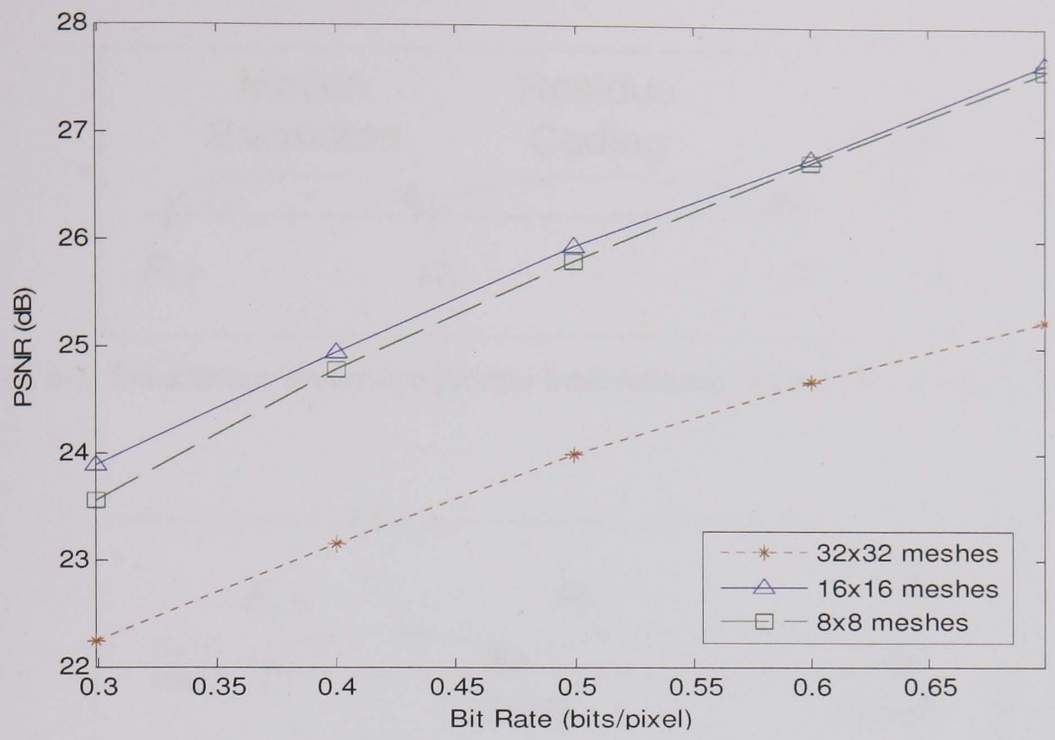


Fig. 6-2 Rate-distortion performance of *Football* sequence employing mesh-coding with different patch sizes.

motion activity. The overall scheme includes a dual layer structure, significance maps at different precisions and a group-wise approach to motion information coding.

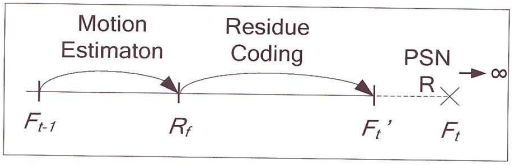


Fig. 6-3 Transitional conversion process from reference frame, $\mathcal{F}_{t-1}$, to target frame, $\mathcal{F}_t$.

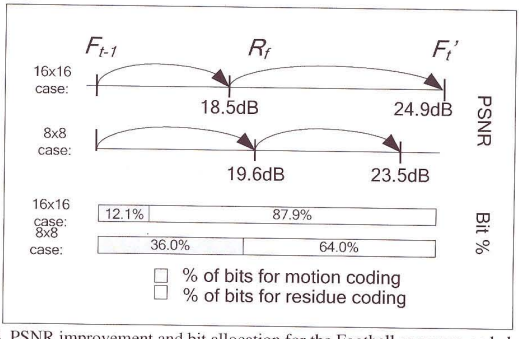


Fig. 6-4 PSNR improvement and bit allocation for the Football sequence coded at 0.30 bits/pixel.

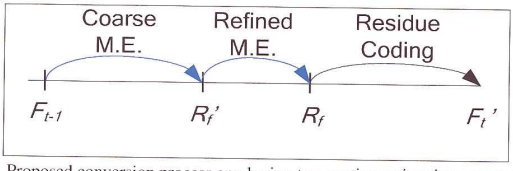


Fig. 6-5 Proposed conversion process employing two motion estimation processes prior to residue coding.

6.1.2 Motion Coding v. Residue Coding

A video coding process can be interpreted as shown in Fig. 6-3, where $\mathcal{F}_t$ and $\mathcal{F}_{t-1}$ are the target and reference frames, respectively. The goal is to convert $\mathcal{F}_{t-1}$ to $\mathcal{F}_t'$, a reconstructed frame resembling $\mathcal{F}_t$, by employing motion compensation and residue coding. The overhead in the process is the encoded motion vectors and frequency spectra derived from the residue data. The limit of effective motion prediction is dependent on the correlation between the reference frame and the current frame. Once the limit is reached, the prediction cannot be improved at the expense of additional bits. In the case of a fixed bit rate, an over-allocation of bits to the motion description has consequences for the residue coding. Fig. 6-4 examines the *Football* sequence coded at 0.30bits/pixel. The diagram shows the PSNR improvement and percentage bit allocation for motion coding and residue coding. One can see that a significant improvement of 1.1dB in motion prediction is achieved by estimating motion with a finer mesh. In contrast, the four times increase in bits assigned to motion information results in a poor performance for the residue coding. This degradation is especially serious for the rate-distortion performance at low bit rates, as reflected in Fig. 6-2. In the example, the overall PSNR is boosted by certain regions having a need for a more detailed motion description. However the additional motion vectors describing static objects are essentially redundant. An attempt to deliver additional bits to predict the movement of new occluded or discovered objects ought to be avoided, as residue coding is a more efficient method of encoding them.

Meshes incorporating different sized patches provide a good solution to the aforementioned problem, since finer meshes need only be used where required. Fig. 6-5 illustrates the concept of two prediction stages, each employing a different mesh density. The utilisation of a finer mesh in the second stage exploits further temporal

redundancy prior to residue coding. The question that arises is how to apply finer meshes efficiently in order to overcome the connectivity constraint. Furthermore, the overhead of specifying the location of regions employing the finer meshes should be reduced to the lowest degree. In the next section, we present a dual-layer approach to fulfil these two objectives. Simulation results for the proposed algorithm integrated in a complete codec are presented in Section 6.3. Finally, conclusions are discussed in Section 6.4.

6.2 Dual Layer Structure for Mesh-based Video Coding

The proposed dual-layer structure [106, 107] predicts the motion of objects using mesh topologies of different densities. In the initial basic layer, a triangular mesh of patch size 16x16 pixels is applied to obtain an intermediate predicted frame, $\mathcal{R}_f'$. Significance maps are subsequently constructed to determine the location of motion-active regions which are further represented by the progressive layer fine mesh of patch size 8x8 pixels. The encoder for the proposed algorithm is illustrated in Fig. 6-6, where the shaded boxes represent the new features introduced in the dual-layer structure. As shown, motion coding is elaborated to three stages prior to group-wise entropy coding. The following subsections describe the operation of each stage.

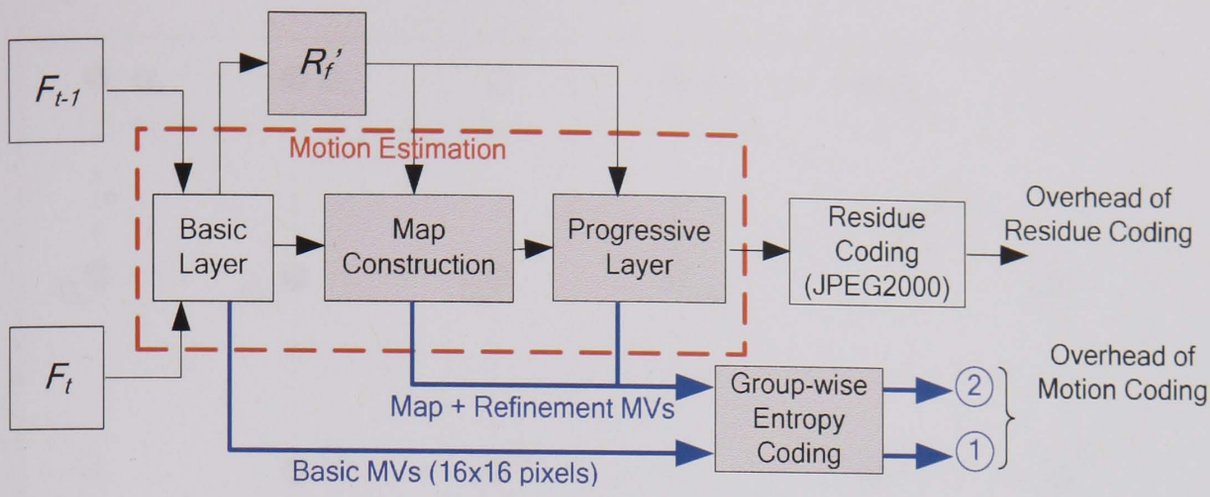


Fig. 6-6. Motion estimation is elaborated to three stages followed by group-wise entropy coding. The shaded boxes represent new features introduced by the proposed dual-layer structure.

6.2.1 Estimation in the Basic Layer

Motion estimation in the basic layer uses the hexagonal matching algorithm (HMA) proposed in [84]. The HMA implements a motion search by forming a hexagon from a grid point's six neighbours as illustrated in Fig. 6-7. The motion vector for the internal grid point, G_z, is obtained by deforming the mesh with an affine transform. Estimation of the motion for each of the other vertices forming the hexagon, G_a to G_f, is performed in a similar manner. Subsequently, a refinement process is applied to each grid point if the displacement of any grid point in the hexagon has been modified. The iterative refinement process is applied to all grid points until there is convergence to local or global minima. Subsequent to the motion estimation, an intermediate frame, $\mathcal{R}_f'$, is restored by deforming the mesh of the reference frame with updated displacements of all the grid points.

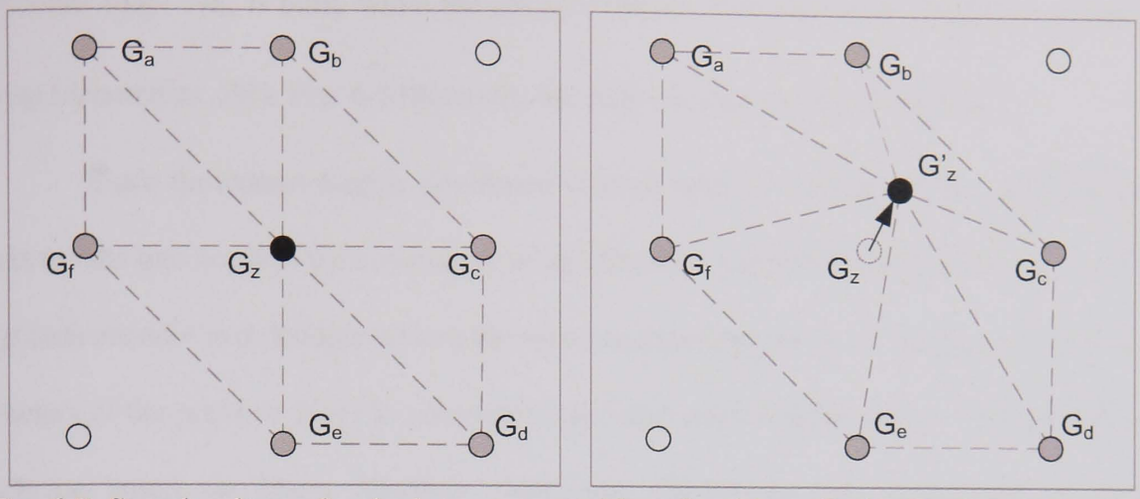


Fig. 6-7 Search scheme for the hexagon matching algorithm: (Left) Fix the position of the six vertices, G_a to G_f ; (Right) Search for best position of G_z

The grid points with non-zero displacement show the existence of motion activity. Intuitively one would expect a better prediction to be achieved by further estimation with a finer mesh. However, this is not always the case. If a rigid movement exists (simple camera panning action), the motion estimation at this stage is sufficient. Another case is the movement of the surrounding grid points resulting from the sudden appearance of an occluded object. As mentioned, the allocation of additional bits in these cases does not necessarily result in a significant improvement in the prediction. Ideally, a finer mesh should be applied to those parts of an object that appear to have disparate motion, for example, at the boundaries of a moving object. In the next subsection, we describe how these areas are detected through the construction of a significance map.

6.2.2 Construction of the Significance Maps

The significance map contains binary decisions as to whether grid points require further motion estimation. The aim of reducing transmission overhead is achieved by constructing maps of different precision at both sides of the codec. At the decoder, a less

accurate map, $\mathcal{M}_c$, is built, while the encoder further elaborates it to produce a refined map (denoted as $\mathcal{M}_R$). Fig. 6-8 illustrates the map construction at the encoder.

Since the coarse map is developed at both sides of the codec, the construction has to take into account the constraints of the decoder. A feasible assumption is made to let both encoder and decoder restore the same intermediate frame by sending the motion vectors of the previous layer in advance of the other motion information. An evaluation scheme based on Mean Absolute Difference (MAD) is then developed for the construction of the coarse map. The proposed scheme calculates a MAD value for a region (of size K-by-L) centred at each grid point in the reference and intermediate frames, as shown in Fig.6-9. Note that the positions of the grid points have been redefined for the intermediate frame, such that an identical coordinate system is established in both frames. The proposed MAD evaluation is recorded as follows:

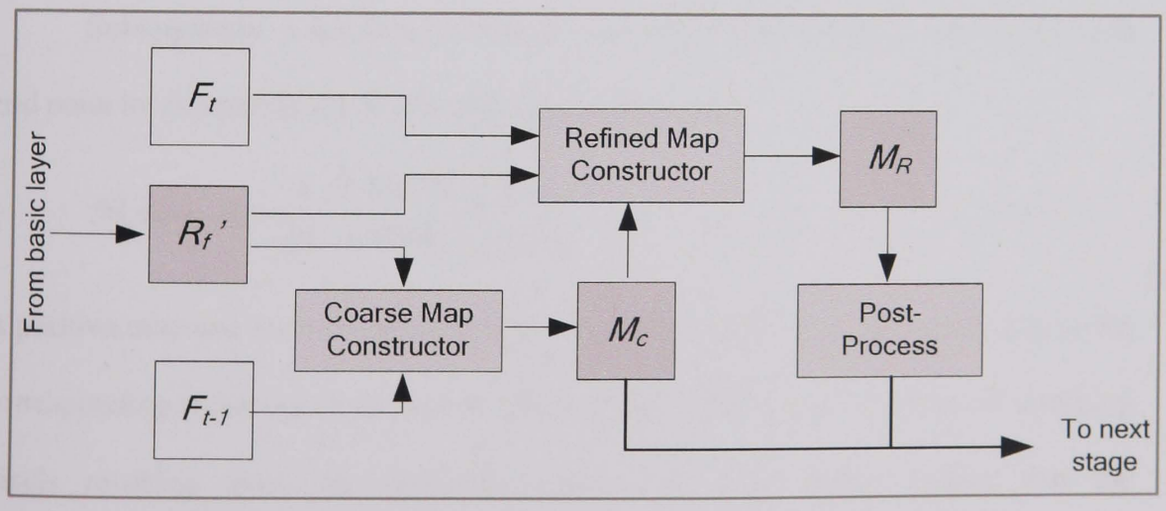


Fig. 6-8 Block diagram showing the construction of significance maps, where M_c and M_R are the coarse and refined maps, respectively.

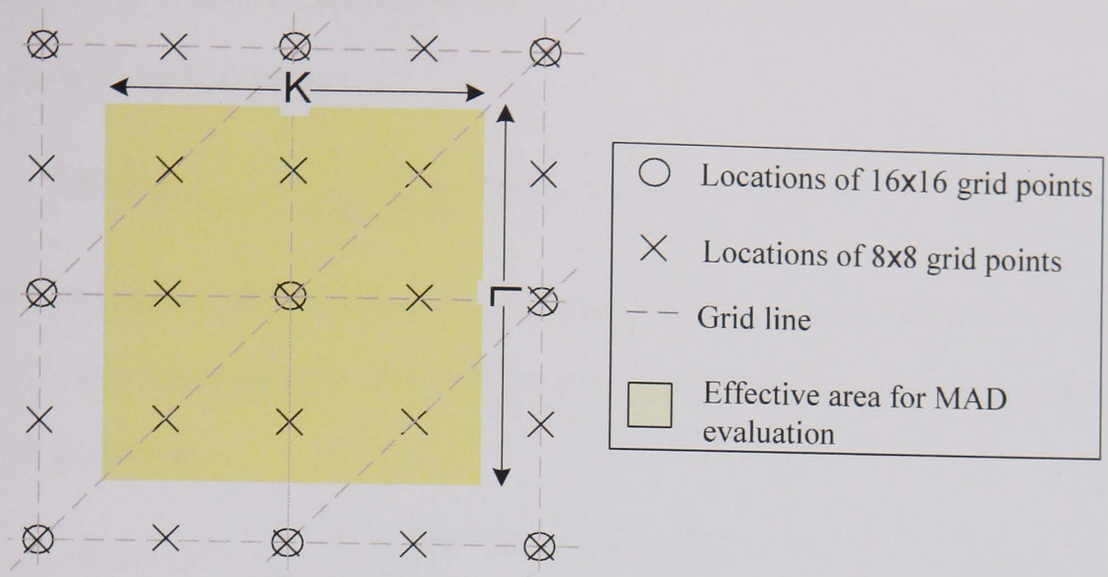


Fig. 6-9 Effective area for modified MAD evaluation in Equation.(6-1) and (6-3) .

$$\mathrm{MAD}_{G(i,j)} = \frac{1}{K \times L} \sum_i \sum_j \left| \tilde{B}_{R'}(G(i,j)) - \tilde{B}(G(i,j), t-1) \right| \qquad (6\text{-}1)$$

where $G(i, j)$ is the grid point at location (i, j); $\tilde{B}_{R'}(G(i,j))$ and $\tilde{B}(G(i,j),t-1)$ are the rectangular areas of size K-by-L centred at $G(i, j)$ in the intermediate frame and previous frame, respectively.

Subsequently, a decision process is used to provide a binary outcome for each grid point by comparing the MAD with a threshold, Δ_{Th1}:

$$\mathcal{M}_c(G(i,j)) = \begin{cases} 1 & \text{if } \mathrm{MAD}_{G(i,j)} \geq \Delta_{Th1} \\ 0 & \text{if } \mathrm{MAD}_{G(i,j)} < \Delta_{Th1} \end{cases} \qquad (6\text{-}2)$$

A positive outcome in (6-2) indicates that a significant intensity difference exists in the corresponding region between the two frames. This is due to the existence of wrapping pixels resulting from the deformed meshes. It may further reflect that the aforementioned regions comprise non-homogenous content. Consequently, an advanced examination of the target frame is performed to determine the content. The process is

similar to that of (6-1) except that the reference frame is replaced by the target frame, $\mathcal{F}_t$. The equation is as follows:

$$\mathrm{MAD}'_{\mathcal{M}_c(G(i,j))=1} = \frac{1}{K \times L} \sum_i \sum_j | \tilde{B}(G(i,j),t) - \tilde{B}_{R'}(G(i,j)) | \qquad (6\text{-}3)$$

Note that Equation (6-3) is selectively performed on the valid grid points that have resulted from Equation (6-2). The decision process for the refined map, M_R, is also altered as follows:

$$\mathcal{M}_R(G(i,j)) = \begin{cases} 1 & \text{if } \Delta_{Th1} \le \mathrm{MAD}'_{\mathcal{M}_c(G(i,j))=1} \le \Delta_{Th2} \\ 0 & \text{otherwise} \end{cases} \qquad (6\text{-}4)$$

One can see that a positive outcome is generated if the MAD' value in (6-3) is between the two thresholds, Δ_{Th1} and Δ_{Th2}. The application of the first threshold removes the regions which have similar content to that of the target frame. In contrast, the application of Δ_{Th2} detects highly inaccurate predictions in a region resulting from the previous layer. One such cause is the appearance of a new occluded object. These regions are not considered for improvement.

Valid candidates in the refined map indicate that the corresponding region requires motion refinement in the next stage. A post-process is still needed to remove solitary candidates from the map. The reason is to prevent 'expensive' bits being consumed by isolated regions which do not necessarily lead to a significant improvement in prediction. Eventually, the contents of both the coarse and refined maps are passed to the progressive layer for motion refinement. This refinement is described in the next subsection.

6.2.3 Refinement in the Progressive Layer

The progressive layer aims to improve regional prediction in the intermediate frame by the enhancement of finer meshes. Motion refinement is performed selectively at the valid grid points outlined in the refined map. However, the significance map is built on a different scale of the coordinate system. Thus, precision conversion is performed by activating the eight adjacent 8x8 grid points surrounding each valid candidate in the refined map, as shown in Fig. 6-10. Note that the same conversion process is also applied to the coarse map to acquire information of the disparities with the refined map. As the disparity regions require transmission without undergoing motion refinement, zero motion is specified. As for other grid points indicated as 'background' in the refined map, they are not permitted to change displacement during the refinement process. This restriction does not violate the application of the hexagon matching algorithm. Subsequent to the refinement process, the coding of updated motion vectors and the map disparities is accomplished with a group-wise coding scheme described in Section 6.2.4.

The proposed algorithm provides a solution to the mesh-connectivity problem described earlier. This is due to the strategy of employing motion searches at two different scales. The first layer corrects overall motion displacement that exists in the reference frame. The moving details in the intermediate frame are further predicted with finer meshes based on a different set of grid points. The proposed scheme results in a more detailed search compared to the application of finer meshes on the reference frame directly. Thus, a better regional prediction is sometimes expected.

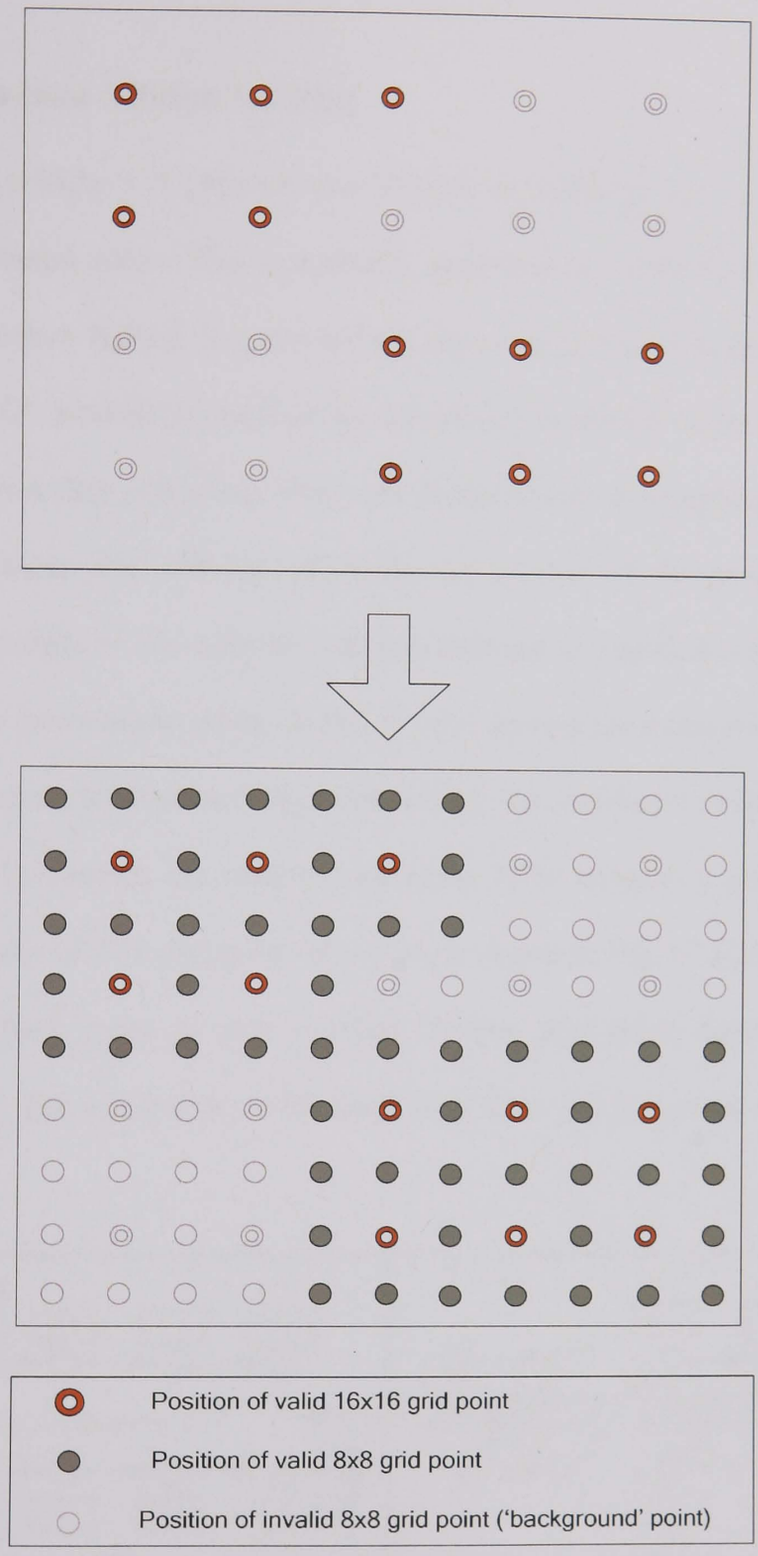


Fig 6-10 The precision conversion between 16x16 and 8x8 grid point coordinate systems.

6.2.4 Group-wise Motion Coding

In mesh-based models, it is observed that clusters of stationary grid points are found when coding natural video. This is normally attributed to a stationary intensity that exists in consecutive frames. It is particularly prevalent in scenes comprising a static background [100]. Intuitively, much of the redundancy in motion vector coding can be eliminated by grouping grid points with zero displacement. A group-wise approach is proposed to achieve this, without specifying the location of the groups. Fig. 6-11 illustrates an example of two different schemes utilising an Exp-Golomb [103] lookup table for motion information coding. In Fig. 6-11(a), motion information is encoded in a conventional raster scan manner without considering the correlation of adjacent motion displacements. In contrast, the proposed algorithm [105] employs a group-wise scan considering classes of four grid points at a time, as shown in Fig. 6-11(b). An indicator is assigned to each group to state whether all four grid points have zero motion (indicator = '0'). For a group in which there is at least one grid point with non-zero

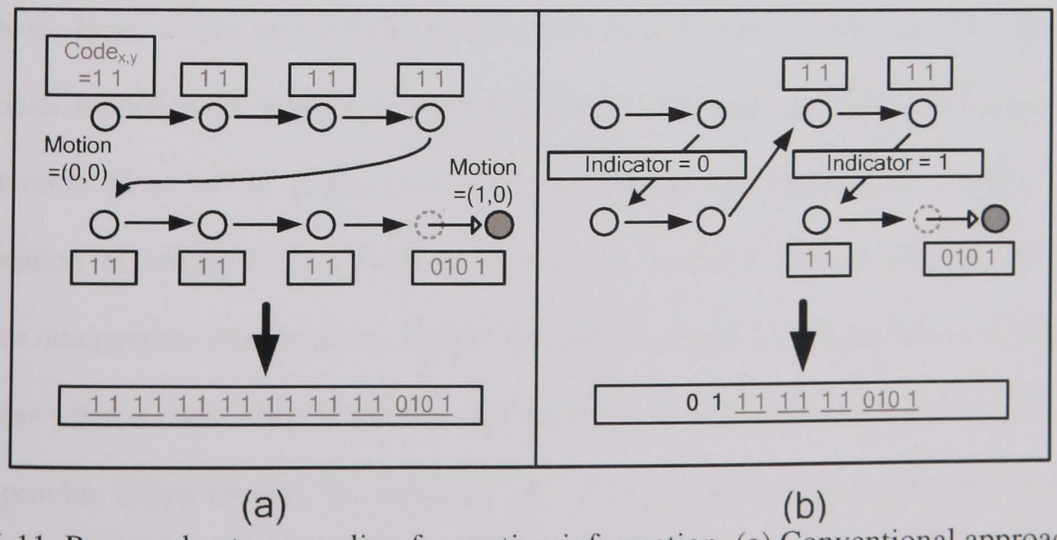


Fig.6-11 Proposed entropy coding for motion information. (a) Conventional approach using raster scan; (b) proposed approach incorporating indicator bits and a group-wise scan resulting in fewer bits to encode the same amount of motion information.

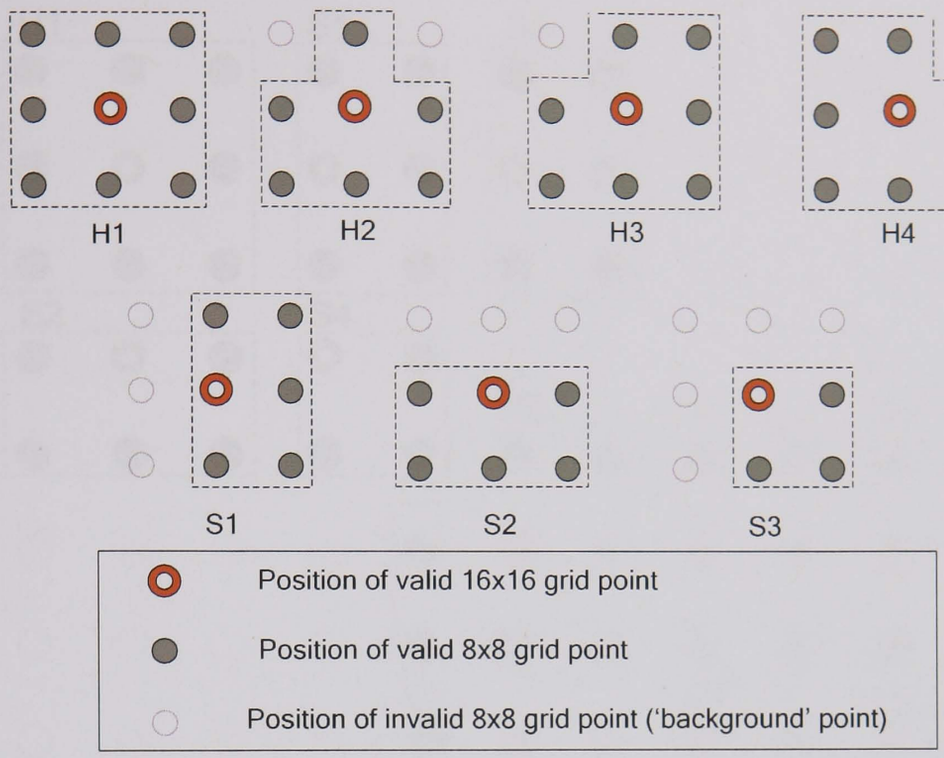


Fig. 6-12 Available pattern descriptions for motion plane in the progressive layer. The top four models (H1 ~ H4) represent the 'header' of an isolated motion-activity region, followed by subsequent models (S1 ~ S3).

motion, the indicator is set to '1' and is followed by the motion vector difference (MVD) code for each member of the group.

The proposed group-wise approach [106] provides better coding efficiency in the basic layer where zero displacements appear in clusters. In contrast, the refined locations defined in the significance map are unlikely to remain stationary. Furthermore, the motion plane in the progressive layer is of irregular shape which violates the application of coding with a fixed group size. A modified scheme featuring group pattern descriptions is proposed to resolve the problem. Fig 6-12 depicts seven available patterns when a raster order is used to obtain groups. The groups in the top row (H1 ~ H4) provide initial choices for collating grid points, followed by subsequent group models (S1 ~ S3) according to the location at which they reside. Fig. 6-13 illustrates the application of the pattern descriptions on an irregularly shaped motion plane in the

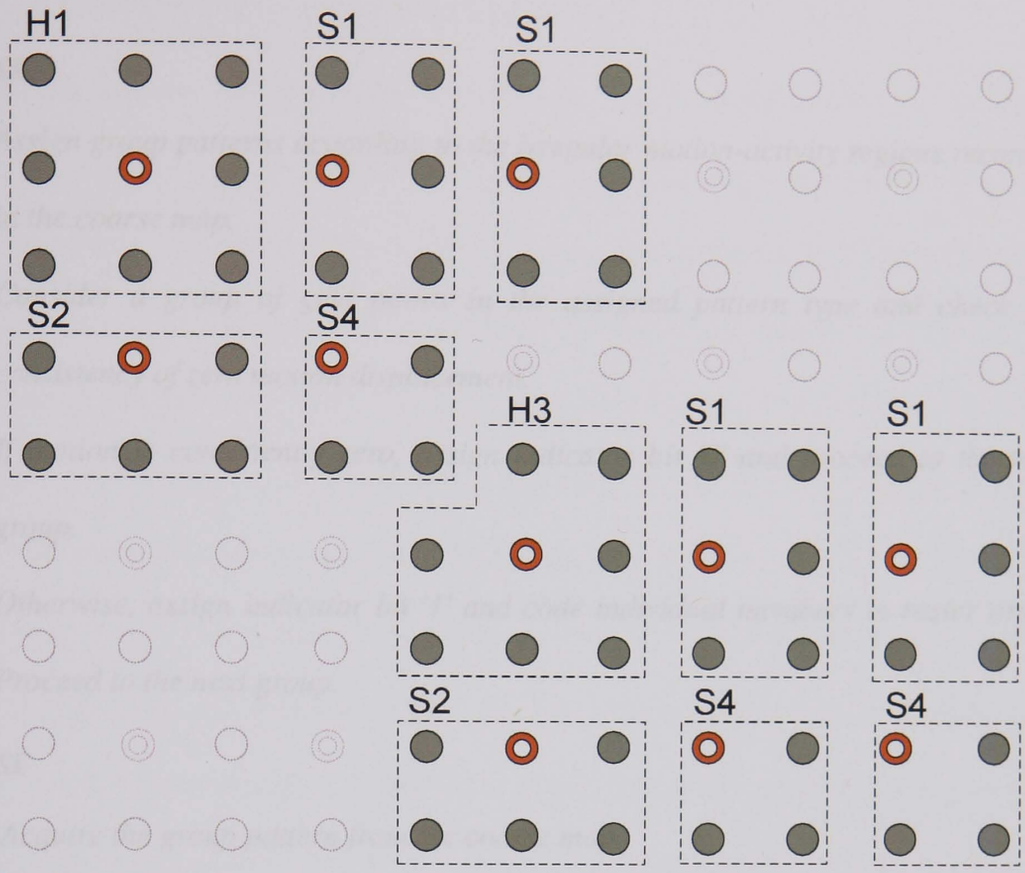


Fig. 6-13 The application of pattern description on motion plane of irregular size in the progressive layer.

progressive layer. The transmission of the pattern descriptions is unnecessary as the decoder is able to obtain the information through the coarse map.

The pattern description approach provides a single goal coding scheme for all information in the progressive layer. The coding information includes the motion plane of the progressive layer, the actual shape of the refined map, as well as the disparity regions (padded with zeros) between the two maps. The efficiency of the proposed scheme is especially beneficial when zero motion vectors are prevalent in the progressive motion plane, as groups of zero motion vectors can be represented by just one indicator bit.

The proposed progressive layer motion coding algorithms for encoder and decoder are summarised as follows:

Encoder:

1. *Assign group patterns according to the irregular motion-activity regions recorded in the coarse map.*
2. *Consider a group of grid points in the assigned pattern type and check for consistency of zero motion displacement.*
3. *If motion is consistently zero, assign indicator bit '0' and proceed to the next group.*
4. *Otherwise, assign indicator bit '1' and code individual members in raster order. Proceed to the next group.*

Decoder:

1. *Acquire the group pattern from the coarse map.*
2. *Examine the indicator bit (first bit of the bit-stream).*
3. *If the indicator bit is '0', assign all members of the group as having zero motion displacement. Return to 2.*
4. *Otherwise, decode the next codewords for individual members of the corresponding group pattern, and return to 2.*

6.2.5 Decoder for the Proposed Algorithm

The decoder follows the procedure described in Fig. 6-14. The received bit-stream comprising both motion information and entropy-coded residue data is decoded on a frame-wise basis. The motion information for each frame can be further decomposed into two layers. The first layer specifies the motion vector differences (MVDs) for the grid points in the reference frame. By deforming the triangular mesh with the updated grid point displacements, an intermediate predicted frame is restored. The decoder repeats the same procedures described in the encoder to create the coarse map. The shape of the map is further refined by the motion information in the second layer. Note that this layer also contains the MVDs for the regional refinement that allows the decoder to modify the details of the moving object. Eventually, a final predicted frame is reconstructed subsequent to the residue compensation process.

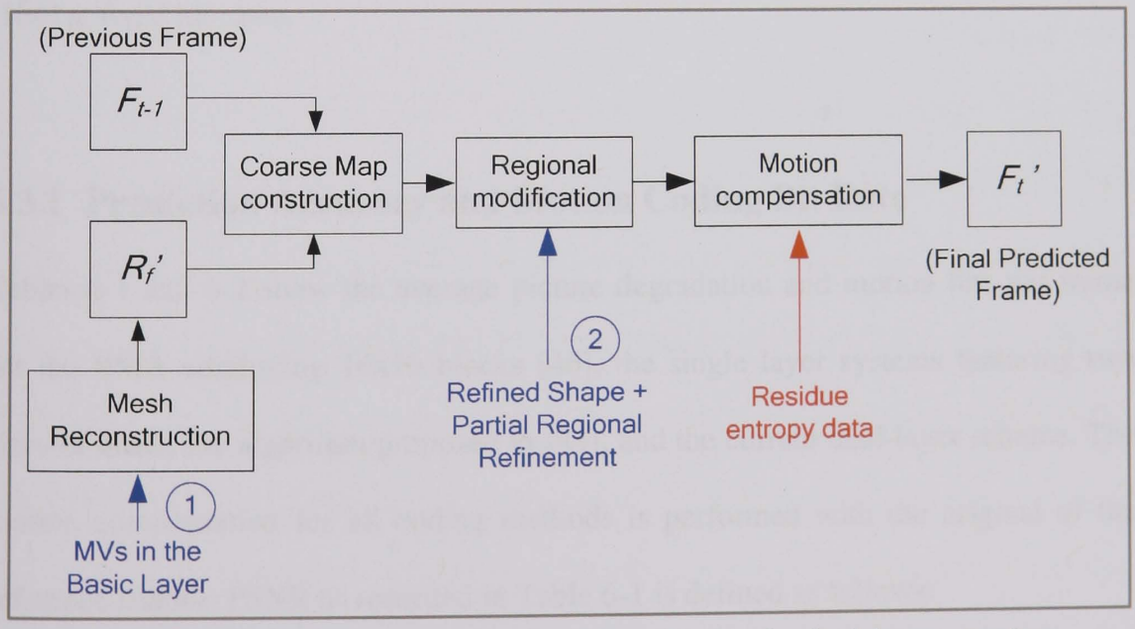


Fig. 6-14 The diagram of the proposed decoder for the dual-layer structure.

6.3 Simulations, Comparisons, and Discussions

This section discusses the performance of the proposed algorithm in a complete codec. Results are presented as improvements over motion compensation employing triangular meshes of different scales and block partitions of size 16x16 pixels. The video sequences selected were of QCIF resolution (176x144), and 30 frames of each sequence were processed. Except for the first frame, all frames are encoded with the hexagonal matching algorithm (HMA) [84] for mesh-based systems, and the block-matching algorithm (BMA) in the case of block-based prediction. The search ranges for motion estimation in the first layer is ±8 pixels, and in the second layer, ±4 pixels, if applicable. The dimensions of MAD evaluation regions in (6-1) and (6-3) are 17x17 pixels. Δ_{Th1} and Δ_{Th2} in (6-4) is an empirically determined constant of 5.0 and 40.0 respectively for all test sequences. Finally, the motion information was in all cases encoded using the Exp-Golomb method [103] and residue data using JPEG 2000 [104] at fixed bit rates.

6.3.1 Prediction Accuracy and Motion Coding Bit Rate

Tables 6-1 and 6-2 show the average picture degradation and motion bits per frame for the BMA employing 16x16 blocks [46], the single layer systems featuring two sizes of mesh, the algorithm proposed in [84], and the current dual-layer scheme. The motion compensation for all coding methods is performed with the original of the reference frames. PSNR as recorded in Table 6-1 is defined as follows:

$$\text{PSNR} = 20.\log_{10}\left(\frac{255\times 255}{\sqrt{\text{MSE}}}\right) \tag{6-5}$$

where MSE represents the mean square error between the original frame $\mathcal{F}_t$ (of size W x L) and the frame reconstructed from motion prediction, $\mathcal{R}_f'$.

$$\text{MSE} = \frac{1}{W \times L} \times \sum_{i=0}^{W-1} \sum_{j=0}^{L-1} \left(\mathcal{F}_t(i,j) - \mathcal{R}_f'(i,j) \right)^2 \qquad (6\text{-}6)$$

Note that both tables show results prior to residue coding with JPEG 2000. The content of sequences in Classes A and B exhibit little motion. In contrast, Class C sequences comprise fast moving objects which are relatively challenging for an efficient motion estimator.

As indicated in Table 6-1, each algorithm provides different degrees of PSNR improvement depending on the sequence class. General tendencies are identified as follows: In Classes A & B, the algorithm employing a global 8x8 mesh obtains the best PSNR performance. This is due to an over-dense population of grid points distributed throughout the entire frame. Similarly, the dual-layer structures performing regional refinement provide a PSNR improvement over a single layer with a 16x16 mesh and the BMA. For Class C sequences, the block-based model provides the most accurate motion prediction in the sequences *Ice Skating* and *Football* that contain predominantly translational motion. However, the performance of the BMA approach degrades when coping with rotational trajectories as in *Fun Fair* and *Mobile & Calendar*. In contrast, the proposed dual-layer structure alleviates the connectivity constraints by allowing effective search areas of ± 8 pixels and ± 4 pixels in the first and second layers, respectively. This has provided a solution to the aforementioned problems in both single-layer strategies. The advantages are evident from Table 1 where the dual-layer structure achieves better results than the BMA in *Fun Fair* and *Mobile & Calendar*, while significant gains over the 8x8 mesh are observed in the *Ice Skating* and *Football* sequences.

Table 6-2 shows the average number of bits required to code the motion information for each of the five algorithms. Bits required for residue coding are not included. In general more bits are required for the single layer 8x8 mesh and for Park et al's dual-layer algorithm [87], because of the increased number of grid points. As expected, an increase in the number of bits leads to better prediction accuracy. However the proposed algorithm that incorporates group-wise coding is effective in addressing this trade-off. The group-wise coding is particularly effective for sequences in Classes A & B where zero displacements are prevalent. For Class C sequences, an increased number of bits for motion coding is unavoidable. However, in general, the proposed scheme requires fewer bits than the single layer 8x8 mesh, for a similar level of prediction accuracy. In conclusion, the proposed algorithm achieves a better compromise between prediction accuracy and bits spent in motion information coding.

6.3.2 Algorithm Complexity

Table 6-3 gives an indication of the algorithm complexity in terms of the number of affine iterations per frame. The evaluation of BMA complexity is excluded from the comparison. Due to the different mesh size employed, the effective area for each algorithm is varied to obtain the best result. For example, the search range for both single layer systems is set at ± 8 pixels, while the dual-layer systems perform motion searches of ± 8 pixels in the basic layer and ± 4 pixels in the progressive layer. In practice, the search area in the progressive layer is one-quarter of that in the system layer with an 8x8 mesh. One would expect this to be reflected in the complexity measurements.

It is observed from Table 6-3 that the single layer system employing an 8x8 mesh requires more affine iterations for completion of the search operation. For the dual-layer systems, the proposed algorithm generally requires less computation for Class A & B sequences compared to Park et al's algorithm. However, the situation is reversed when encoding Class C sequences. This explains the reason for the increased PSNR values recorded in Table 6-1. Comparing the proposed algorithm with the single layer 8x8 system, significantly fewer affine iterations are performed yet a better or similar PSNR performance is achieved. This is illustrated in the *Football* sequence where a total of only 1117 iterations are required compared with 1406 for the single 8x8 mesh, and the PSNR (Table 6-1) is increased from 19.64dB to 20.68dB.

6.3.3 Rate-distortion Performance

Fig. 6-15 to Fig. 6-20 illustrates the PSNR v. bit-rate performance for six sequences *Hall Monitor* (Class A), *Silent Voice* (Class B), *Fun Fair*, *Ice Skate*, *Mobile & Calendar*, and *Stefan* (all Class C), for the proposed algorithm and the other four approaches. The results are obtained after residue coding with JPEG 2000 at fixed bit rates. The conversion table for the two popular bit rate measurements for a QCIF picture decoded at 30Hz is shown in Table 6-4.

Table 6-4
The conversion table for the two popular bit rate measurements for a QCIF picture decoded at 30Hz

.Bits per pixel	Kbits per second
0.10	74.25
0.15	111.38
0.20	148.50
0.25	185.63
0.30	222.75
0.40	297.00
0.50	371.25
0.60	445.50

Table 6-1 Average PSNR per frame prior to the residue coding

Class A & B	16x16 block	16x16 mesh	8x8 mesh	Dual Layer in [87]	**Proposed**
Coastguard	32.67 dB (+1.55dB)	33.97 dB (+0.25dB)	34.82 dB (-0.60dB)	34.31 dB (-0.09dB)	**34.22 dB**
Hall Monitor	37.70 dB (+0.71dB)	38.01 dB (+0.40dB)	38.89 dB (-0.48dB)	38.54 dB (-0.13dB)	**38.41 dB**
News	38.68 dB (+0.54dB)	38.81 dB (+0.41dB)	39.90 dB (-0.68dB)	39.26 dB (-0.04dB)	**39.22 dB**
Silent	37.52 dB (+1.12dB)	37.41 dB (+1.23dB)	38.70 dB (-0.06dB)	38.83 dB (-0.09dB)	**38.64 dB**
Suzie	38.38 dB (+0.12dB)	38.49 dB (+0.01dB)	38.99 dB (-0.49dB)	38.64 dB (-0.14dB)	**38.50 dB**
Class C	16x16 block	16x16 mesh	8x8 mesh	Dual Layer in [87]	**Proposed**
Carphone	32.73 dB (+0.22dB)	32.52 dB (+0.43dB)	33.44 dB (-0.49dB)	33.03 dB (-0.08dB)	**32.95 dB**
Football	20.82 dB (-0.14dB)	18.51 dB (+2.17dB)	19.64 dB (+1.04dB)	20.08 dB (+0.60dB)	**20.68 dB**
Fun Fair	24.31 dB (+1.22dB)	24.10 dB (+1.33dB)	25.46 dB (-0.03dB)	25.22 dB (+0.21dB)	**25.43 dB**
Ice Skate	25.51 dB (-0.28dB)	22.34 dB (+2.89dB)	23.83 dB (+1.40dB)	24.47 dB (+0.76dB)	**25.23 dB**
Mobile	25.74 dB (+0.27dB)	25.85 dB (+0.16dB)	26.11 dB (-0.10dB)	26.01 dB (+0.00dB)	**26.01 dB**
Stefan	25.29 dB (+1.16dB)	24.99 dB (+1.46dB)	26.05 dB (+0.40dB)	26.11 dB (+0.34dB)	**26.45 dB**

(xx.xx dB) = PSNR of the proposed dual-layer – PSNR of other algorithm

Table 6-2 Average Motion Bits per frame prior to the residue coding

Class A & B	16x16 block	16x16 mesh	8x8 mesh	Dual Layer in [87]	**Proposed**
Coastguard	248.21 bits	257.17 bits	987.24 bits	526.68 bits	**346.13 bits**
Hall Monitor	216.83 bits	205.03 bits	1145.10 bits	414.31 bits	**152.59 bits**
News	208.90 bits	205.31 bits	880.00 bits	276.85 bits	**102.06 bits**
Silent	237.72 bits	227.37 bits	962.68 bits	389.13 bits	**215.96 bits**
Suzie	220.14 bits	222.14 bits	972.97 bits	352.85 bits	**107.51 bits**
Class C	16x16 block	16x16 mesh	8x8 mesh	Dual Layer in [87]	**Proposed**
Carphone	358.83 bits	373.45 bits	1558.97 bits	762.64 bits	**346.13 bits**
Football	895.93 bits	708.62 bits	2754.41 bits	1726.88 bits	**2793.02 bits**
Fun Fair	348.21 bits	358.13 bits	1418.82 bits	975.20 bits	**949.61 bits**
Ice Skate	431.72 bits	406.83 bits	1587.24 bits	1109.37 bits	**1166.24 bits**
Mobile	217.86 bits	214.68 bits	958.76 bits	337.30 bits	**109.40 bits**
Stefan	348.28 bits	326.96 bits	1393.79 bits	780.47 bits	**771.28 bits**

Table 6-3 Average Complexity (affine iterations) per frame

Class A & B	16x16 mesh	8x8 mesh	Dual Layer in [87]	**Proposed**
Coastguard	264.38 [a]	1110.45 [b]	264.38 [a] + 139.00 [c]	**264.38 [a] + 15.48 [c]**
Hall Monitor	149.79 [a]	717.72 [b]	149.79 [a] + 70.07 [c]	**149.79 [a] + 14.97 [c]**
News	130.14 [a]	522.72 [b]	130.14 [a] + 25.13 [c]	**130.14 [a] + 16.21 [c]**
Silent	149.00 [a]	626.76 [b]	149.00 [a] + 51.21 [c]	**149.00 [a] + 40.90 [c]**
Suzie	150.45 [a]	655.59 [b]	150.45 [a] + 56.00 [c]	**150.45 [a] + 0.93 [c]**
Class C	16x16 mesh	8x8 mesh	Dual Layer in [87]	**Proposed**
Carphone	274.38 [a]	1237.41 [b]	274.38 [a] + 146.55 [c]	**274.38 [a] + 73.65 [c]**
Football	476.82 [a]	1406.34 [b]	476.82 [a] + 299.72 [c]	**476.82 [a] + 640.48 [c]**
Fun Fair	300.51 [a]	1130.21 [b]	300.51 [a] + 235.10 [c]	**300.51 [a] + 291.97 [c]**
Ice Skate	332.34 [a]	1101.59 [b]	332.34 [a] + 204.24 [c]	**332.34 [a] + 239.38 [c]**
Mobile	140.96 [a]	626.14 [b]	140.96 [a] + 51.76 [c]	**140.96 [a] + 49.83 [c]**
Stefan	292.03 [a]	1015.55 [b]	292.03 [a] + 190.21 [c]	**292.03 [a] + 251.79 [c]**

[a] 16x16 mesh with ±8 search range; [b] 8x8 mesh with ±8 search range; [c] 8x8 mesh with ±4 search range

Generally, all other algorithms obtain a PSNR gain over the single layer 8x8 mesh at low bit rates, where the 8x8 system suffers from insufficient bits available for residue coding. However, the PSNR-rate curve of the 8x8 mesh overtakes that of the 16x16 mesh for the Class C sequences as the bit rate increases. With regard to the block-based model, the PSNR gain over the 16x16 mesh is strongly correlated to the prediction accuracy presented in Table 6-1. The coding gains are especially significant in sequences containing predominantly translational motion, such as *Ice Skating* and *Stefan*. In contrast, the proposed algorithm provides a steady PSNR gain over each of the other algorithms at most bit rates. This is attributed to the fact that the proposed dual-layer structure obtains the maximum prediction accuracy but uses fewer bits for motion coding. This is especially true for most sequences when coded at low bit rates. A PSNR improvement ranging from 0.2dB to 1.5dB is found between the proposed algorithm and the least performing algorithm in all cases.

6.4 Summary

New applications, such as mobile multimedia and video streaming over low-bandwidth channels, necessitate improvements in low bit-rate coding. The MPEG-x and H.26x motion compensation schemes can provide visually unacceptable results when insufficient bits are spent in residue coding. The main reason is attributed to the employment of the Block Matching Algorithm (BMA), which fails to provide an effective description for non-translational motion. In contrast, the use of mesh-based motion prediction provides an alternative when coping with various types of spatial motion, such as translation, rotation, and zoom. This is credited to the fact that the moving object is described with patches deformed by affine transformation in the

connectivity constraint. The motion is defined by the change in position of the corresponding vertices (grid points). However, the computation of the motion vector for a grid point is affected by the interdependencies among its neighbours. To that degree, studies based on search strategies, using an optimisation framework, and employing a close-form solution were proposed respectively. Those approaches were briefly discussed.

In the chapter, we also examined the rate-distortion performance of the sequences employing different patch sizes. A study regarding the correlation between motion coding and residue coding was detailed to provide an explanation of the phenomenon. A dual-layer approach incorporating different sized patches was then proposed to improve the coding efficiency of the mesh-based motion compensation. The algorithm initially applies meshes of size 16x16 pixels to identify the moving objects in a picture. By constructing a significance map at both sides of the codec, the overhead of specifying exact locations for the finer mesh is removed. A finer mesh is then used selectively to compensate for any inaccurate predictions in the previous layer. Finally, a group-wise coding scheme is incorporated to exploit the redundancy of motion vectors in both layers.

In Section 6.3, simulations employing an extensive choice of sequences are presented to obtain measurements of prediction accuracy, bits spent on motion information coding, and the complexity of the algorithm. The results indicate that the proposed algorithm obtains a better compromise between prediction accuracy and the number of bits expended on motion description. PSNR-rate diagrams for Class C sequences confirm the efficiency of the technique for a wide range of video coding rates.

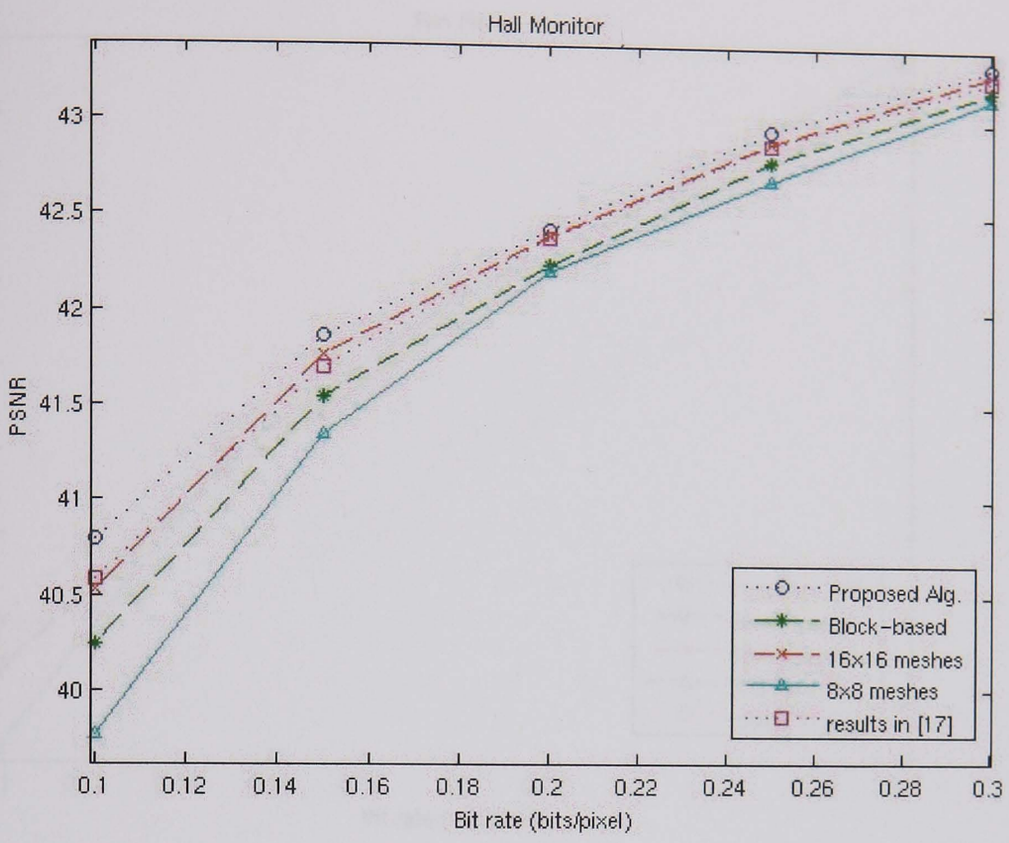


Fig. 6-15 PSNR-rate diagrams for *Hall Monitor* in Class A

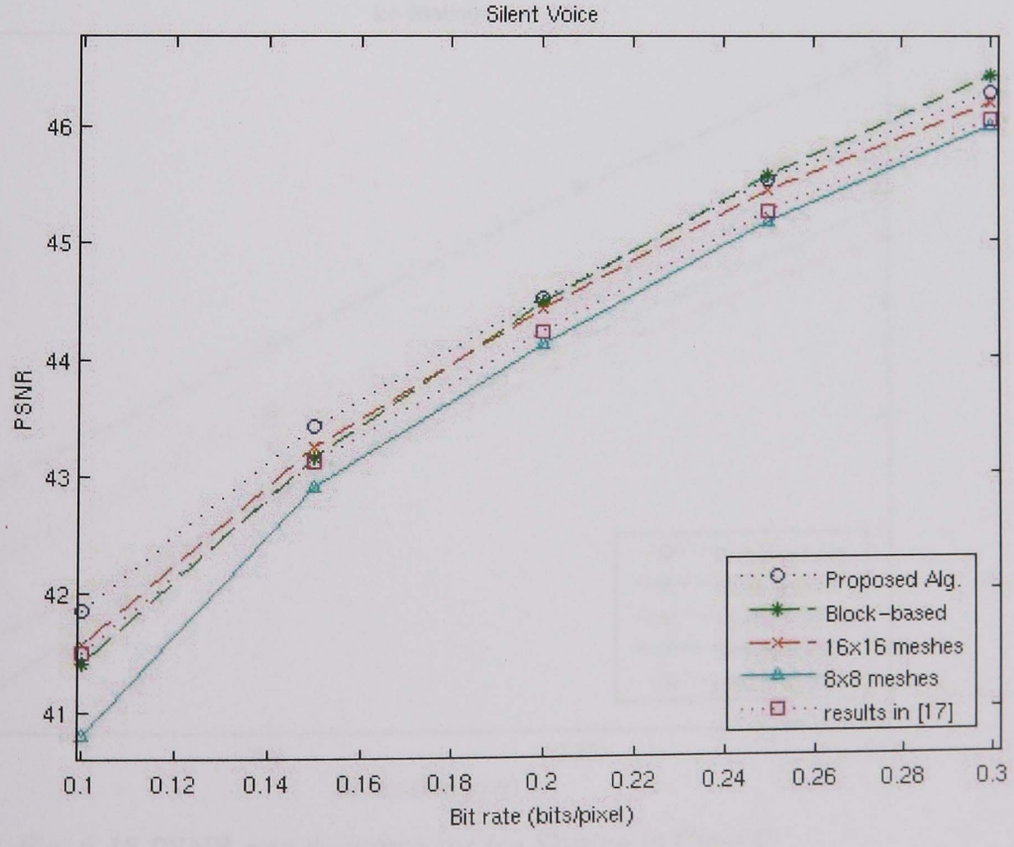


Fig. 6-16 PSNR-rate diagrams for *Silent Voice* in Class B

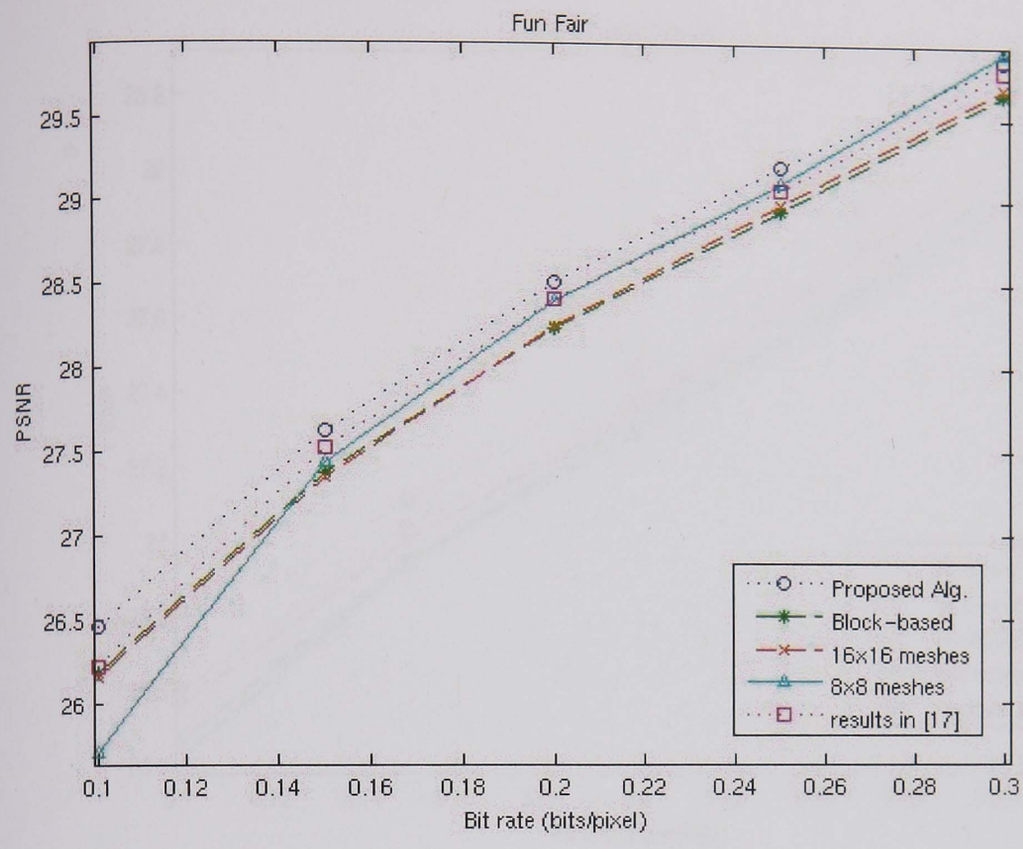


Fig. 6-17 PSNR-rate diagrams for *Fun Fair* in Class C

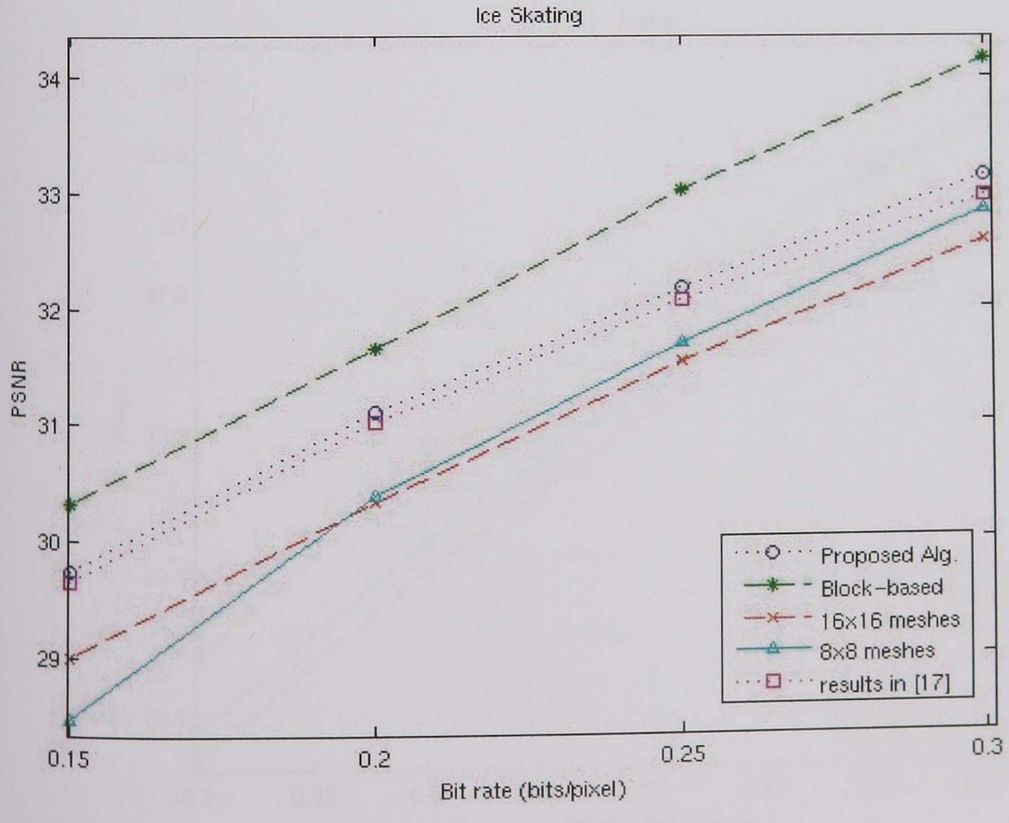


Fig. 6-18 PSNR-rate diagrams for *Ice Skating* in Class C

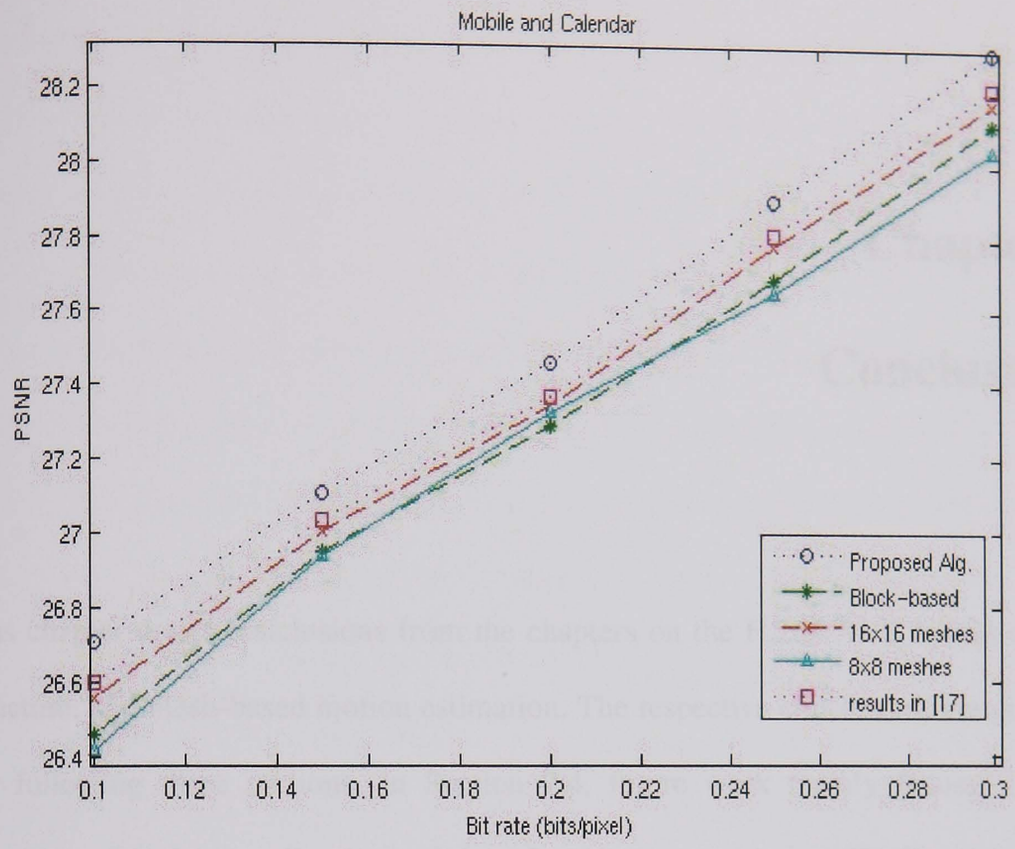


Fig. 6-19 PSNR-rate diagrams for *Mobile and Calendar* in Class C

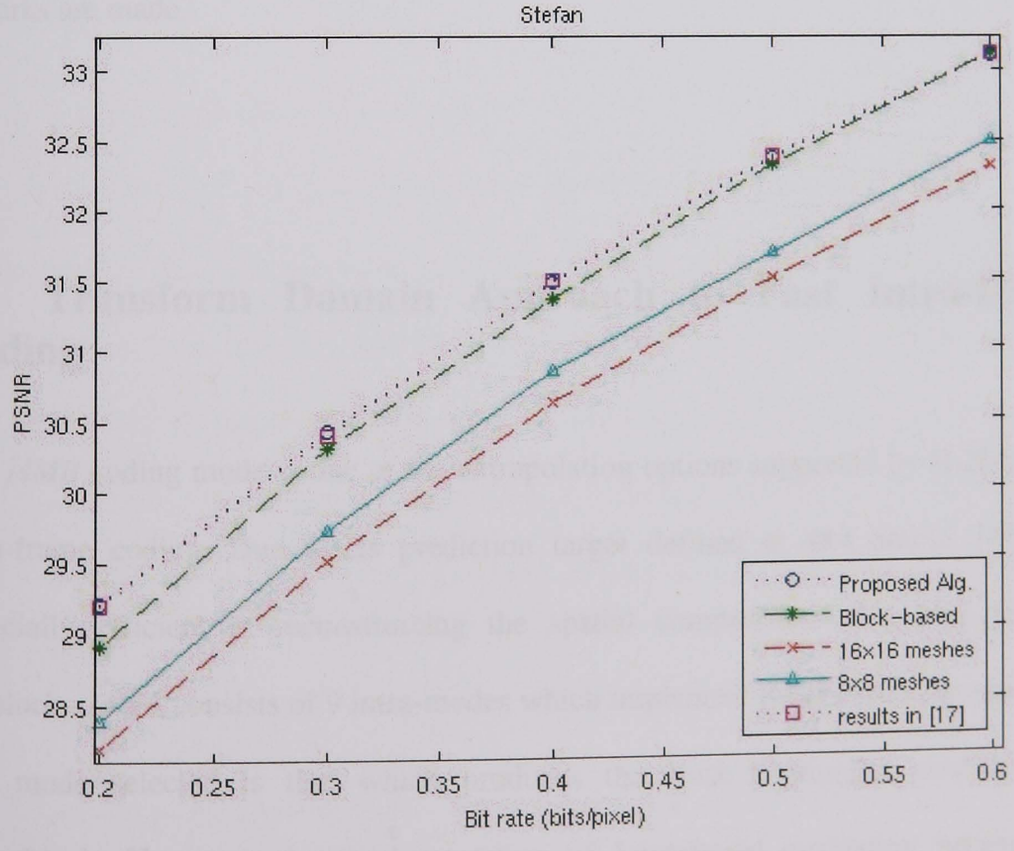


Fig. 6-20 PSNR-rate diagrams for *Stefan* in Class C

Chapter 7

Conclusions

This chapter draws conclusions from the chapters on the H.264/AVC hybrid coding structure, and mesh-based motion estimation. The respective conclusions are given in the following three sections. In Section 7.4, future work mainly focuses on the extension of scalable coding in mesh-based motion compensation. Finally, concluding remarks are made.

7.1 Transform Domain Approach to Fast Intra-frame Coding

The *I4MB* coding mode is one of the extrapolation options supported by H.264/AVC intra-frame coding. Due to its prediction target defined at 4x4 pixels, *I4MB* is especially efficient in reconstructing the spatial composition of a high-detailed subblock. *I4MB* consists of 9 intra-modes which implement pixel-based extrapolation. The mode selected is that which produces the least Lagrangian cost for the macroblock. However, the implementation of Lagrangian evaluation necessitates valuable computational resources due to the requirements of image reconstruction and

the estimation of bits spent for each intra-mode. To reduce the computational requirements imposed by the H.264/AVC mode selection process, several spatial domain approaches have been developed. Chapter 3 concentrates on the transform domain approach originally introduced in [80, 83]. The nature of the technique is a fast algorithm operating in the IntDCT domain to achieve efficient computation for intra-frame coding using the *I4MB* mode. Furthermore, the proposed IntDCT domain algorithm exhibits some important advantages over spatial domain approaches:

- The proposed algorithm employs the inherent frequency characteristic of an original block and its predicted block without any priori knowledge required by the spatial domain approaches, such as a predefined threshold and other priori macroblock information.
- Even though operating in the IntDCT domain, the complete computation of all frequency spectra is not required.
- The extrapolation processes are only performed for the modes that require further Lagrangian examination.

The core feature of the proposed *Fintra* algorithm is the employment of matrix formulae for the pre-selection process. Without significantly increasing the computational overhead, the algorithm intelligently selects fewer candidate members (say M in number) required for the second round selection using a more expensive Lagrangian evaluation. In later experiments, it is determined that the minimum value of M should be 3 in order to maintain a better compromise between rate and distortion performance.

The *Fintra* algorithm is integrated into the coding process of JM6.1e, a benchmark codec recommended by JVT. The performance is evaluated using various test sequences featuring a wide range of texture composition and motion activity.

Each test sequence, comprising 50 frames, is encoded at five different compression levels. The simulation compares the performance with that of Pan et al's algorithm [64], one of the best-performing spatial domain approaches in the literature. The results shown in the rate-distortion relationship diagrams indicate that the proposed *Fintra* algorithm halves the PSNR degradation caused by Pan et al's algorithm. In terms of speedup, the proposed transform domain algorithm is verified to gain additional speedup over the best-performing spatial domain approach by approximately 7% on average. In conclusion, the simulations show that the proposed *Fintra* algorithm achieves the same coding performance in terms of picture quality and compression ratio as that of the H.264/AVC standard, yet reduces the computational requirement by up to 57%.

7.2 Multiple Mode Selection for H.264/AVC Inter-Frame Coding

H.264/AVC inter-frame coding incorporates motion estimation and rate-distortion optimisation to provide compression in a temporal fashion. Since each macroblock could be coded differently as an intra-block or an inter-block, inter-modes featuring multiple partitions, as well as other hybrid coding options (such as SKIP and intra-coding options), are taken into consideration during the mode decision process. Thus, the transform domain approach developed in Chapter 3 can be integrated into the H.264/AVC hybrid coding scheme.

In Chapter 4, two fast implementations, namely the *Finter1* [42] and *Finter2* [50] algorithms for H.264/AVC inter-frame coding are described. The success of the two algorithms is achieved by the coding structures comprising several hierarchical decisions/levels. By discarding the least possible block size, both algorithms provide different levels of early termination for the mode selection process. Mode knowledge of the previously encoded frame(s) is employed by the proposed *Finter1* algorithm, whereas the *Finter2* algorithm incorporates temporal similarity detection and the detection of different moving features within a macroblock. Despite the difference in decision structure, both coding schemes make use of a spatial complexity measurement to determine the content of the macroblocks.

The details of the two proposed algorithms are roughly described as follows. In the *Finter1* algorithm, the macroblocks are assigned with one of three mode categories according to their texture constitution, determined by the outcome of spatial complexity measurement. The higher detailed the macroblocks are, the more prediction modes the proposed algorithm has to check. In contrast, the *Finter2* algorithm introduces additional measurements targeted at two kinds of encoded

macroblocks: (a) macroblocks encoded with the *SKIP* mode (direct copy from the collocated macroblock); (b) macroblocks encoded by the inter-frame modes with larger decomposed partition sizes (greater than 8×8 pixels). By successfully identifying these two kinds of macroblocks, the encoder is not required to examine them with all possible inter-frame modes, which saves encoding time.

The performance of both algorithms is evaluated using various test sequences in three sequence classes, each of which is separately integrated into the codec of the JM6.1e benchmark. The experimental simulations examine the picture quality and bit rate for five compression levels. The results indicate that the *Finter2* algorithm performs better than the *Finter1* method in all respects. However, a marginal PSNR difference (at most 0.07dB) is noticed when a comparison is made between both algorithms at most fixed bit-rates. With regard to speedup, computational reductions by employing both algorithms are in evidence. On average, a saving of 47% in encoding time is achieved for the *Finter2* algorithm and 33% for the *Finter1* algorithm.

The *Finter1* and *Finter2* algorithms are shown to provide a speedup for inter-frame coding. However, the performance gains of both algorithms were dependent on the spatial content and amount of motion in the video sequence, for a limited range of compression levels. Motivated by this situation, an improved scheme is proposed in Chapter 5. The improved scheme, denoted as the *IFinter* algorithm, contains a more thorough search process and more robust predictions in selecting an optimal inter-mode for each macroblock. With regard to its hierarchical decision structure, the *IFinter* algorithm is considered as an extension of the previously proposed *Finter2* algorithm.

The *IFinter* coding structure is described as comprising three levels. The first level of the proposed algorithm identifies the macroblocks to be encoded with *SKIP*. A detection strategy incorporating an adaptive threshold and a fast transform-quantised implementation was then developed. Unlike the threshold approach described in the *Finter2* method, the improved scheme accurately detects the skipped-macroblock without the need of an increase in computation. The core technique is the computation of a transform-quantised threshold for each IntDCT coefficient at low frequencies.

In the second level, the spatial complexity measurement is still acquired to determine the texture detail of a macroblock. However, the *IFinter* scheme takes into consideration the case of high detailed macroblocks that can be adequately described by large partitions. By computing the RD cost of a few inter-modes, the search process continues if the best mode for a high-detailed macroblock shows favour to be encoded with partition sizes of 16x8 and 8x16. Otherwise, further examination is not required. With regard to the final level, a sophisticated search process is recorded. The need for coding with smaller partition inter-modes is determined by the unequal condition utilising RD cost.

The performance of the proposed *IFinter* coding scheme is compared to that of the previously reported *Finter2* algorithm and Wu et al's method [76], one of the best performing fast algorithms in the literature. The experiments focus on the coding of Class C sequences at a wide range of compression levels. The PSNR-rate relationship diagrams demonstrate that the proposed *IFinter* scheme significantly improves the *Finter2* algorithm in all respects. With regards to the computational saving, the *IFinter* algorithm provides the best result. This is especially true for Class C sequences where the new scheme performs twice as well as the other two algorithms.

In the second simulation, the *Fintra* algorithm for intra-block coding is integrated into a complete hybrid structure of the fast H.264/AVC implementation. The experimental outcomes confirm that the combined algorithm provides an extra 15% saving in computation. More importantly, the combined algorithm ensures at least a 50% reduction in computation compared with the JM6.1e implementation, without degradation in any objective measurement. This is achieved regardless of the sequence class and any coding condition.

7.3 Mesh-based Motion Compensation Featuring A Dual-Layer Structure

The dual-layer coding technique employing a mesh-based model is another topic detailed in the thesis. The use of mesh-based motion prediction provides an alternative when coping with various spatial deformations, such as translation, rotation, and zoom. This is credited to the fact that the moving object is described by patches that undergo affine transformation. Compared with a disjointed collection of blocks in BMA, mesh-based motion models provide more visually acceptable results. In Chapter 6, the rate-distortion performance employing different patch sizes was examined. A study regarding the correlation between motion coding and residue coding was conducted to provide explanations. A dual-layer approach incorporating different sized patches was then proposed to improve the coding efficiency.

The proposed mesh-based coding structure [106, 107] is described as comprising two prediction stages, each employing a different mesh density. The algorithm initially applies meshes of size 16x16 pixels (in the basic layer) to identify the moving objects in a picture. By constructing a significance map at both sides of the codec, the overhead of specifying exact locations for the finer mesh is removed. A finer mesh of size 8x8 pixels (in the progressive layer) is then used selectively to compensate for any inaccurate predictions in the basic layer.

Despite the proposed coarse-to-fine motion prediction, a group-wise coding scheme [105] is incorporated to exploit the redundancy of motion vectors in both basic and progressive layers. The motion vector coding for the basic layer employs a group-wise scan considering groups of four grid points at a time. An indicator is assigned to each group to state whether all four grid points have zero motion. With regards to the progressive layer, a modified scheme [106] featuring group pattern descriptions is

proposed to resolve the constraint of an irregular shape motion plane. It also provides a single goal coding scheme for all information in the progressive layer.

The performance of the proposed dual-layer structure is examined extensively by means of experimental simulations. The simulations employing three classes of sequence are presented in terms of prediction accuracy, bits spent on motion information coding, and the complexity of the algorithm. The results indicate that the proposed algorithm obtains a better compromise between prediction accuracy and the number of bits expended on the motion description. PSNR-rate diagrams for Class C sequences confirm the efficiency of the technique for a wide range of video coding rates.

7.4 Further Work

Future work will focus on the extension of the scalability feature for the proposed dual-layer coding structure in Chapter 6. Scalable coding is a technique which enables an encoder to arrange the coded bit stream in a number of layers, including a base layer and several enhancement layers (Fig 7-1). The decoder can optimise the video quality over a given bit rate range by selecting part of the coded bitstream. A basic quality sequence is obtained if a decoder selects to receive the base layer bitstream, whereas, a higher quality sequence is presented if more enhancement layers are received as well as the base layer. Some outstanding issues are briefly introduced for the incorporation of mesh-based motion compensation.

Scalability on Mesh-based Motion Compensation

The scalable coding structure in Fig. 7-1 has several similarities to that illustrated in the proposed mesh-based structure of Fig. 6-5. First and foremost, the data for transmission is coded in a multi-layer fashion. Subsequently, both encoders utilise motion estimation at different precisions to generate the data stream required in each layer. Last but not the least, the motion information derived from the coarse layer in

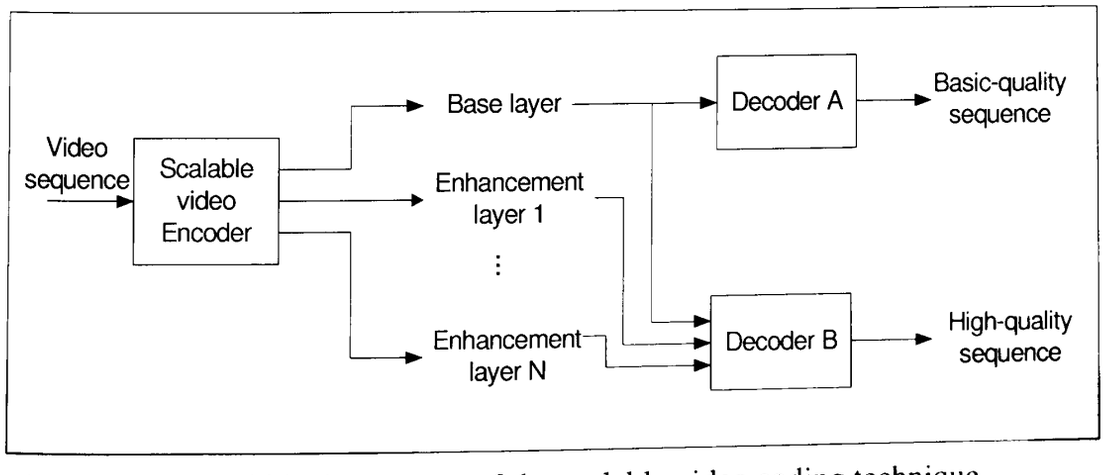


Fig. 7-1 General block structure of the scalable video coding technique.

Fig. 6-5 is actually part of the base layer in Fig. 7-1. The same relationship is applied to the progressive layer and the enhancement layers. Based on the aforementioned observations, one can see the connection between the previously proposed algorithm and its potential scalable feature. However, several technical issues have to be taken into consideration:

- The design of the scalable system for a mesh-based model, e.g., motion estimation for the reconstructed frames in the form of layers.
- The automatic detection of unnatural artefacts caused by the partial improvement in a picture. It requires a design of a regional filter to cope with this.
- The scheme of priority arrangement for transmitting motion information, e.g., enhanced motion information for the current frame or basic motion information for the next frame.
- Furthermore, entropy coding featuring unequal protection and regions of interest in a picture are other interesting topics to be investigated.

A careful design of the coding scheme would provide a scalable system to selectively transmit different parts of bit streams according to the available bandwidth of the connection.

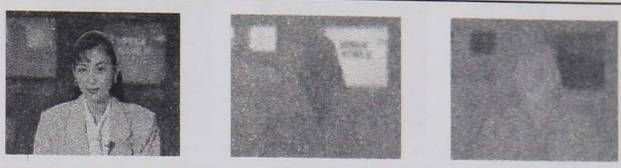
7.5 Concluding Remarks

This thesis developed several improved algorithms for both block-based and mesh-based hybrid video coding. In the block-based system, achievement mainly focuses on the design of the fast implementation for the H.264/AVC video coding standard. Chapters 3, 4, and 5 analyse the problems of inefficient computation addressed by the intra-frame coding and inter-frame coding. The improvements for each coding scheme were detailed. The performance of the proposed algorithms has been evaluated extensively in terms of picture degradation, compression performance, PSNR-rate relationship diagram, computational speedup and subjective examination. Overall, the new technique demonstrates the same coding performance in terms of the picture quality and compression ratio as that of the H.264/AVC standard, yet achieves a saving in encoding time of up to 77%. This is true regardless of the video content and the coding conditions.

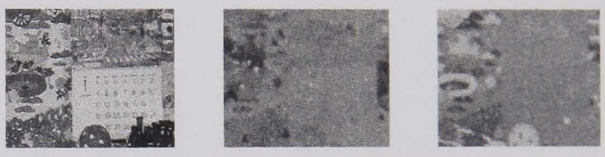
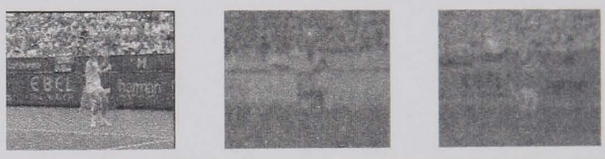
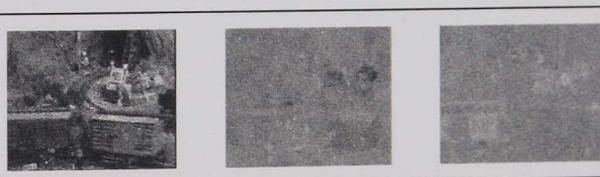
In Chapter 6, a dual-layer motion compensation technique for mesh-based video coding that addresses the deficiencies introduced by single fixed patch sizes was developed. The problem of coding the motion overhead and residue data were examined in detail. The dual-layer approach was then developed to overcome the constraints of connectivity and coding overhead. Overall, the proposed dual-layer structure was shown to achieve significant improvements of up to 2dB over traditional mesh-based coding at fixed bit-rates.

Appendix 1

Image Test Sequences

Class	Sequences	Sample (Y, Cr, Cb)
A	Akiyo	
	Container	
	Grandma	
	Hall Monitor	
	Miss America	

Class	Sequences	Sample
B	City	
	Coastguard	
	Foreman	
	News	
	Paris	
	Silent Voice	
	Suzie	

Class	Sequences	Sample
C	Football	
	Fun Fair	
	Harbour	
	Ice Skating	
	Mobile and Calendar	
	Stefan	
	Tempete	
	Train and Tunnel	
	Table Tennis	
	Waterfall	

Appendix 2 The coefficients of Ω_{mode} at various frequency positions

$$\Omega_{DC} = \begin{array}{c} \begin{array}{cccccccccccc} A & B & C & D & E & F & G & H & I & J & K & L & M \end{array} \\ \begin{pmatrix} 16 & 16 & 16 & 16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 16 & 16 & 16 & 16 & 0 \\ 8 & 8 & 8 & 8 & 0 & 0 & 0 & 0 & 8 & 8 & 8 & 8 & 0 \\ 1 & 4 & 8 & 12 & 14 & 12 & 8 & 5 & 0 & 0 & 0 & 0 & 0 \\ 12 & 8 & 4 & 1 & 0 & 0 & 0 & 0 & 12 & 8 & 4 & 1 & 14 \\ 15 & 13 & 10 & 3 & 0 & 0 & 0 & 0 & 6 & 3 & 1 & 0 & 13 \\ 13 & 9 & 1 & 0 & 0 & 0 & 0 & 0 & 8 & 9 & 10 & 3 & 11 \\ 3 & 10 & 15 & 16 & 13 & 6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 9 & 15 & 34 & 0 \end{pmatrix} \end{array} \begin{array}{l} \text{VERT} \\ \text{HORT} \\ \text{DC} \\ \text{DIAG\_DL} \\ \text{DIG\_DR} \\ \text{VERT\_R} \\ \text{HORT\_D} \\ \text{VERT\_L} \\ \text{HORT\_U} \end{array}$$

(A-1)

$$\Omega_{AC\,(0,1)} = \begin{array}{c} \begin{array}{cccccccccccc} A & B & C & D & E & F & G & H & I & J & K & L & M \end{array} \\ \begin{pmatrix} 32 & 16 & -16 & -32 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 7 & 10 & 7 & 0 & -7 & -10 & -9 & 0 & 0 & 0 & 0 & 0 \\ -7 & -10 & -7 & -2 & 0 & 0 & 0 & 0 & 7 & 10 & 7 & 2 & 0 \\ 5 & -16 & -17 & -6 & 0 & 0 & 0 & 0 & 10 & 6 & 2 & 0 & 16 \\ -5 & -4 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 7 & 5 & -2 \\ 6 & 17 & 14 & -6 & -18 & -11 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 6 & 2 & -13 & 0 \end{pmatrix} \end{array} \begin{array}{l} \text{VERT} \\ \text{HORT} \\ \text{DC} \\ \text{DIAG\_DL} \\ \text{DIG\_DR} \\ \text{VERT\_R} \\ \text{HORT\_D} \\ \text{VERT\_L} \\ \text{HORT\_U} \end{array}$$

(A-2)

$$\Omega_{AC\,(1,0)} = \begin{array}{c} \begin{array}{cccccccccccc} A & B & C & D & E & F & G & H & I & J & K & L & M \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 32 & 16 & -16 & -32 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 7 & 10 & 7 & 0 & -7 & -10 & -9 & 0 & 0 & 0 & 0 & 0 \\ 7 & 10 & 7 & 2 & 0 & 0 & 0 & 0 & -7 & -10 & -7 & -2 & 0 \\ -2 & -2 & 7 & 5 & 0 & 0 & 0 & 0 & -3 & -5 & -2 & 0 & 2 \\ 9 & 6 & 2 & 0 & 0 & 0 & 0 & 0 & 6 & -14 & -17 & -6 & 14 \\ 5 & 7 & 2 & 0 & -5 & -7 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 15 & 11 & -37 & 0 \end{pmatrix} \end{array} \begin{array}{l} \text{VERT} \\ \text{HORT} \\ \text{DC} \\ \text{DIAG\_DL} \\ \text{DIG\_DR} \\ \text{VERT\_R} \\ \text{HORT\_D} \\ \text{VERT\_L} \\ \text{HORT\_U} \end{array}$$

(A-3)

$\Omega_{AC}(2,0) =$

A	B	C	D	E	F	G	H	I	J	K	L	M	
0	0	0	0	0	0	0	0	0	0	0	0	0	VERT
0	0	0	0	0	0	0	0	0	0	0	0	0	HORT
0	0	0	0	0	0	0	0	0	0	0	0	0	DC
4	12	9	-12	-26	-12	9	16	0	0	0	0	0	DIAG_DL
12	-9	-12	-4	0	0	0	0	12	-9	-12	-4	26	DIG_DR
19	9	-9	-10	0	0	0	0	-6	-10	-4	0	11	VERT_R
4	-1	-4	0	0	0	0	0	9	4	-9	-10	7	HORT_D
10	9	-13	21	-1	12	4	0	0	0	0	0	0	VERT_L
0	0	0	0	0	0	0	0	10	7	-13	-3	0	HORT_U

(A-4)

$\Omega_{AC}(2,0) =$

A	B	C	D	E	F	G	H	I	J	K	L	M	
0	0	0	0	0	0	0	0	0	0	0	0	0	VERT
0	0	0	0	0	0	0	0	16	-16	-16	16	0	HORT
0	0	0	0	0	0	0	0	0	0	0	0	0	DC
1	2	0	-2	-2	-2	0	3	0	0	0	0	0	DIAG_DL
-2	0	2	1	0	0	0	0	-2	0	2	1	-2	DIG_DR
-1	-3	0	1	0	0	0	0	0	1	1	0	1	VERT_R
-5	-3	1	0	0	0	0	0	0	1	4	3	-1	HORT_D
1	0	-1	0	-1	0	1	0	0	0	0	0	0	VERT_L
0	0	0	0	0	0	0	0	3	3	-7	0	0	HORT_U

(A-5)

$\Omega_{AC}(0,2) =$

A	B	C	D	E	F	G	H	I	J	K	L	M	
16	-16	-16	16	0	0	0	0	0	0	0	0	0	VERT
0	0	0	0	0	0	0	0	0	0	0	0	0	HORT
0	0	0	0	0	0	0	0	0	0	0	0	0	DC
1	2	0	-2	-2	-2	0	3	0	0	0	0	0	DIAG_DL
-2	0	2	1	0	0	0	0	-2	0	2	1	-2	DIG_DR
-7	-3	4	3	0	0	0	0	2	3	1	0	-3	VERT_R
-1	1	1	0	0	0	0	0	0	-1	0	1	-1	HORT_D
3	4	-5	-8	1	4	1	0	0	0	0	0	0	VERT_L
0	0	0	0	0	0	0	0	1	1	-1	0	0	HORT_U

(A-6)

Bibliography

[1] M. Ghanbari, Standard codecs: image compression to advanced video coding, The Institution of Electrical Engineers, 2nd Edition, 2003.

[2] M. Steliaros, "Motion compensation for 2D object-based video coding," PhD thesis, Department of Computer Science, Warwick University, England, 1999.

[3] M. Kunt, A. Ikonomopoulos, and M. Kocher, "Second-generation image-coding techniques," *Proceedings of the IEEE*, vol. 73, no. 4, pp, 549-574, 1985.

[4] J. Jain and A. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Transaction on Communications*, vol. 29, no. 12, pp. 1799-1808, 1981

[5] S. Kappagantula and K. Rao, "Motion compensated predictive coding," *Proc. of international technical symposium (SPIE) 83*, Aug. 1983.

[6] H. Bergmann, "Displacement estimation based on the correlation of image segments," *IRE conference on the Electronic Image Processing*, July, 1982.

[7] T. Ishiguro and K. Iinuma, "Television bandwidth compression transmission by motion-compensated interframe coding," *IEEE Communication Magazine*, vol. 10, pp. 24-30, 1982

[8] ITU-T Recommendation H.261, "Video codec for audiovisual services at p x 64 Kbits/s," ITU, Geneva, 1993.

[9] ISO/IEC 11172-2, "Information technology – coding of moving pictures and associated audio for digital storage media at up to 1.5 *Mbits/sec* video," ISO/IEC JTC1/SC19 WG11, Dec. 1991.

[10] ISO/IEC 13818-2, "Information technology – generic coding of moving pictures and associated audio – Part 2: video," ISO/IEC JTC1/SC19 WG11, Nov. 1993.

[11] ITU-T and ISO/IEC JTC1, "Generic coding of moving pictures and associated audio information – Part 2: video," ITU-T recommendation H.262 and ISO/IEC 13818-2 (MPEG-2), Nov. 1994.

[12] ITU-T Recommendation H.263, "Video coding for low bit rate communication," ITU, Geneva, Nov. 1995.

[13] ITU-T Recommendation H.263+, "Video coding for low bit rate communication," ITU, Geneva, Jan. 1998.

[14] ITU-T Recommendation H.263++, "Video coding for low bit rate communication," ITU, Geneva, Feb. 2000.

[15] ISO/IEC FCD 14496-2, "Information technology – coding of audio-visual objects: visual", ISO/IEC JTC1/SC19 WG11, Jan. 1999.

[16] H. Musmann, M. Hötter, and J. Ostermann, "Object-oriented analysis-sythesis coding of moving images," *Signal Processing: Image Communication*, vol.1 no.2, pp.117-138, Oct. 1989.

[17] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini and G. Sullivan, "Rate-constrained coder control and comparison of video coding standards," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 13, pp. 688-703, July 2003.

[18] D. Alfonso, D. Bagni, D. Pau and A. Chimienti, "A Performance analysis of H.264 video coding standard," in *Proc. 23rd Picture Coding Symposium*, Saint-Malo, France, pp. 23-28, April 2003.

[19] A. Joch, F. Kossentini and P. Nasiopoulos, "A Performance analysis of the ITU-T draft H.26L video coding standard," in *Proc. 12th International Packet Video Workshop,* Pittsburgh, U.S.A., April 2002.

[20] G. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, pp. 74-90, Nov. 1998.

[21] A. Jain, "Image data-compression: a review," in *Proc. of the IEEE*, vol. 69, no.3, pp. 349-389, 1981.

[22] H. Musmann, P. Pirsch, and H. Grallert, "Advances in picture coding," in *Proc. of the IEEE*, vol. 73, no. 4, pp.523-548, 1985.

[23] N. Ahmed, T. Natarjan, and K. Rao, "Discrete cosine transform," *IEEE Transaction on Computers*, vol. 40, pp.90-93, 1974

[24] A. Jain, "Fundamentals of digital image processing," Prentice Hall, 1989.

[25] A. Jain, "A sinusoidal family of unitary transforms," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 1, pp. 356-365, Oct. 1979.

[26] W. Chen, C. Smith, and S. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Transaction on Communication*, vol.25, pp.1004-1009, 1979.

[27] V. Bhaskaran and K. Konstandinides, "Image and video compression standards: algorithms and architectures," Kluwer Academic Publishers, 2nd edition, 1997.

[28] T. Koga, K. Iimuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," in *Proc. of National Telecommunications Conference*, pp. 1599-1603, Dec. 1981.

[29] D. Huffman, "A method for the construction of minimum redundancy codes," in *Proc. of the Institute Radio Engineers (IRE)*, vol. 40, no. 9, pp. 1098-1101, 1952

[30] G. Longdon, "An introduction to arithmetic coding," *IBM Journal of Research and Development*, vol. 28, no.2, pp.135-149, 1984.

[31] I. Witten, R. Neal, J. Cleary, "Arithmetic coding for data compression," in *Communications of the ACM*, vol.30, no.6, pp.520-540, 1987.

[32] M. Ghanbari, "Arithmetic coding with limited past history," *IEE Electronics Letters*, vol.27, no.13, pp.1157-1159, 1991.

[33] __, "Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 144496-10 AVC)," Joint Video Team of ISO/IEC and ITU-T, March 2003

[34] A. Hallapuro, M. Karczewicz, and H. Malvar, "Low complexity transform and quantization," in Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Docs. JVT-B038 and JVT-B039, Jan. 2002.

[35] A. Hallapuro and M. Karczewicz,"Low complexity (I)DCT," in Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Docs.VCEG-N43, Sep. 2001.

[36] H. Malvar, "Low-complexity length-4 transform and quantization with 16-bit arithmetic," in Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Docs. VCEG-N44, Sep. 2001.

[37] G. Bjøntegaard, "Calculation of average PSNR differences between RD curves," in Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Docs. VCEG-M33, Mar. 2001.

[38] T. Suzuki and Y. Yagasaki, "Proposal for support CAVLC for 8x8 transform," JVT-L036r1, 5pp., Jul. 2004.

[39] H. Malvar, A. Hallapuro, M. Karczewicz, and L. Kerofsky, "Low-complexity transform and quantization in H.264/AVC," *IEEE Transaction on circuits and systems for video technology*, vol. 13, no. 7, pp. 598-603, Jul. 2003.

[40] G, Malvar, "Signal processing with lapped transforms," Boston, MA: Artech House, 1992.

[41] G. Sullivan, P. Topiwala, and A. Luthra, "The H.264/AVC advanced video coding standard: overview and introduction to the fidelity range extensions," in *Proceedings of SPIE – Applications of Digital Image Processing XXVII*, vol. 5558, pp. 454-474, Aug. 2004.

[42] A. Yu, "Efficient Block Size Selection Algorithm for INTER frame Coding in H.264/AVC," in *Proc. of 29th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 04*, vol. 3, pp. 169-172, Montreal, Canada, May 2004.

[43] D. Marpe, H. Schwarz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 620-636, Jul. 2003.

[44] G. Sullivan, "The H.264/MPEG-4 Advanced Video Coding (AVC) standard," presentation for ITU-T VICA Workshop, downloaded from http://www.itu.int/ITU-T/worksem/vica/docs/presentations

[45] W. Chao, T. Chen, Y. Chang, C. Hsu, and L. Chen, "Computationally Controllable Integer, Half, and Quarter-pel Motion Estimator for MPEG-4 Advanced Simple Profile," in *Proc. of IEEE Int. Symposium on Circuits and System (ISCAS 03)*, 2003, pp. 788–791.

[46] Y. Huang, C. Chen, C. Tsai, C. Shen, and L. Chen, "Survey on block matching motion estimation algorithm and architectures with new results," *Journal of VLSI Signal Processing*, vol. 42, pp. 297-320, 2006

[47] MPEG-4 Video Group, "MPEG-4 video verification model version 11.0," ISO/IEC JTC1/SC19 WG11, Tokyo, March 1998.

[48] W. Pratt, "Spatial transform of colour images," IEEE Transaction on Communication, vol. 19, pp. 980-992, 1971

[49] CCIR Recommendation 601, "Digital methods of transmitting television information," Recommendation 601, 1986.

[50] A. Yu and G. Martin, "Advanced Block Size Selection Algorithm for INTER frame Coding in H.264/AVC," in *Proc. of 11th IEEE International Conference on Image Processing (ICIP) 04,* vol. 1, pp.95-98, Singapore, Oct 2004.

[51] R. Schäfer, T. Wiegand, and H. Schwarz, "The emerging H.264/AVC standard," *European Broadcasting Union (EBU) Technical review*, 12pp., Jan. 2003.

[52] P. Topiwala, G. Sullivan, A. Joch, and F. Kossentini, "Performance evaluation of H.26L TML 8 vs. H.263++ and MPEG-4," presented in 15th Joint Video Team Meeting (VCEG-O42), Pattaya, Thailand, Dec. 2004.

[53] T. Halbach, "Performance comparison: H.26L intra coding vs. JPEG2000," presented in 4th Joint Video Team Meeting, Klagenfurt, Austria, Jul. 2002.

[54] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. Sullivan, "Rate-constrained coder control and comparison of video coding standards," *IEEE Transaction on Circuits and Systems for Video Technology*, vol.13, no. 7, pp. 587-597, Jul. 2003.

[55] M. Horowitz, A. Joch, F. Kossentini, and A. Hallapuro, "H.264/AVC baseline profile decoder complexity analysis," *IEEE Transaction on Circuits and Systems for Video Technology*, vol.13, no. 7, pp. 704-716, Jul. 2003.

[56] JVT reference software, downloaded from http://hhi.de/~suehring/

[57] B. Meng and O. Au "Fast intra-prediction mode selection for 4x4 blocks in H.264," in *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 2003*, vol. 3, pp. 389-392, Hong Kong, Apr. 2003.

[58] B. Meng, O. Au, C. Wong, and H. Lam, "Efficient intra-prediction mode selection for 4x4 blocks in H.264," in *Proc. of International Conference on Multimedia and Expo (ICME) 2003*, vol. 3, pp. 521-524, Baltimore, U.S.A., Jul. 2003.

[59] C. Chen and T. Chang, "Fast three step intra prediction algorithm for 4x4 blocks in H.264," in *Proc. of IEEE International Symposium on Circuits and Systems (ISCAS) 2005*, vol. 2, pp. 1509-1502, Kobe, Japan, May. 2005.

[60] Y. Lin and T. Chang, "Fast block type decision algorithm for intra prediction in H.264 FRext," in *Proc. of IEEE Conference on Image Processing (ICIP) 2005*, vol. 1, pp. 585-588, Singapore, Oct. 2005.

[61] F. Pan, X. Lin, S. Rahardja, K. Lim et al., "Fast mode decision for intra prediction," presented at the 7th Joint Video Team Meeting (JVT-G013), Pattaya, Thailand, Mar. 2003.

[62] F. Pan, X. Lin, S. Rahardja, K. Lim, and Z. Li, "A directional field intra mode decision algorithm for H.264 video coding," in *Proc. of International Conference on Multimedia and Expo (ICME) 2004*, vol. 3, pp. 1147-1150, Taiwan, Jul. 2004.

[63] F. Pan, X. Lin, S. Rahardja, K. Lim et al, "Fast intra mode decision algorithm for H.264/AVC video coding," in *Proc. of IEEE Conference on Image Processing (ICIP) 2005*, vol. 1, pp. 781-784, Singapore, Oct. 2005.

[64] F. Pan, X. Lin, S. Rahardja, K. Lim et al, "Fast mode decision algorithm for intraprediction in H.264/AVC video coding," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 15, no. 7, pp. 813-822, Jul. 2005.

[65] Q. Liu, R. Hu, L, Zhu, X. Zhang, and Z. Han, "Improved fast intra prediction algorithm of H.264/AVC," *Journal of Zhejian University Science A*, vol. I, pp.01-105, 2006.

[66] C. Kim, H. Shih, and C. Kuo, "Feature-based intra-prediction mode decision for H.264," in *Proc. of IEEE Conference on Image Processing (ICIP) 2005*, vol. 1, pp. 769-772, Singapore, Oct. 2005.

[67] C. Kim, H. Shih, and C. Kuo, "Fast H.264 intra-prediction mode selection using joint spatial and transform domain features," *Journal of Visual Communication and Image Representation*, vol. 17, Issue 2, pp. 291-310, Apr. 2006.

[68] J. Xin and A. Vetro, "Fast mode decision for intra-only H.264/AVC coding," in *Proc. of Picture Coding Symposium (PCS) 2006*, 5pp., Beijing, China, Apr. 2006.

[69] J. Lee and B Jeon, "Fast mode decision for H.264," in *Proc. of IEEE International Conference on Multimedia and Expo (ICME) 2004*, vol. 2, pp. 27-30, Jun. 2004.

[70] I. Choi, J. Lee and B Jeon, "Efficient coding mode decision in MPEG-4 part - 10 AVC/H.264 main profile," in *Proc. of International Conference on Image Processing (ICIP) 2004*, vol. 2, 1141-1145, Oct. 2004.

[71] G. Kim, Y. Moon and J. Kim, "An early detection of all-zero DCT block in H.264," in *Proc. of International Conference on Image Processing (ICIP) 2004*, vol. 1, 453-456, Oct. 2004.

[72] Y. Moon, G. Kim and J. Kim, "An improved early detection algorithm for all-zero blocks in H.264 video encoding," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 15, no. 8, pp. 1053-1057, Aug. 2005.

[73] C. Kannangara, I. Richardson, M. Bystrom et al., "Low-complexity skip prediction for H.264 through Lagrangian cost estimation," *IEEE Transaction. on Circuits and Systems for Video Technology*, vol. 16, no. 2, pp. 202-208, Feb. 2006.

[74] X. Jing and L. Chau, "Fast approach for H.264 inter mode decision," *IEE Electronics Letters*, vol. 40, no. 17, Aug. 2004.

[75] M. Yang and W. Wang, "Fast macroblock mode selection based on motion content classification in H.264/AVC," in *Proc. of International Conference on Image Processing (ICIP) 2004*, vol. 2, 741-744, Oct. 2004.

[76] D. Wu, F. Pan, K. Lim et al., "Fast intermode decision in H.264/AVC video coding," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 15, no. 7, pp. 953-958, Jul. 2005.

[77] K. Lim, S. Wu, D. Wu et al., "Fast inter mode selection," ISO/IEC JTC1/SC29/WG11 and ITU-T SG16, Input document JVT I-020, Sep. 2003.

[78] C. Grecos and M. Yang, "Fast inter mode prediction for P slices in the H.264 video coding standard," *IEEE Transaction on Broadcasting*, vol. 51, no. 2, Jun. 2005.

[79] H. Malvar, M. Karczewicz, and L. Kerofsky, "Low complexity transform and quantization in H.264/AVC," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 598-603, Jul. 2003.

[80] A. Yu, G. Martin, and H. Park, "A Frequency Domain Approach to Intra Mode Selection in H.264/AVC," in *Proc. of 13th European Signal Processing Conference (EUSIPCO) 05*, 4pp., Antalya, Turkey, Sep. 2005

[81] A. Yu, G. Martin, and H. Park, "Fast Inter-mode Selection in the H.264/AVC Standard Using Hierarchical Decision Process," under review by *IEEE Transaction on Circuits and Systems for Video Technology*, 10pp.

[82] A. Yu, G. Martin, and H. Park, "Improved schemes for inter-frame coding in the H.264/AVC standard," in *Proc. of International Conference on Image Processing (ICIP) 2005*, vol. 2, 902-905, Sep. 2005.

[83] A. Yu, K. Ngan, and G. Martin, "Efficient intra- and inter-mode selection algorithms for H.264/AVC," *special issue of Journal of Visual Communication and Image Representation*, vol. 17, no. 2, pp. 322-343., Apr. 2006.

[84] Y. Nakaya and H. Harashima, "Motion compensation based on spatial transformations," *IEEE Transaction on Circuit and System for Video Technology*, vol. 4, no. 3, pp. 339-367, Jun 1994.

[85] A. Nosratinia, "New kernels for fast mesh-based motion estimation," *IEEE Transaction on Circuit and System for Video Technology*, vol. 11, no. 1, pp. 40-51, Jan 2001.

[86] Y. Altunbasak and M. Tekalp, "A hybrid video codec with block-based and mesh-based motion compensation modes," *Special Issue of International Journal of Imaging System and Technology*, vol. 9, no.4, pp. 248-256, Aug. 1998.

[87] H. Park, A. Yu and G. Martin, "Progressive mesh-based motion estimation using partial refinement," in Proc. of International Workshop on Very Low Bit-rate Video (VLBV) 2005, 4 pp., Sep. 2005.

[88] N. Božinović, J. Konrad, T. André, M. Antonini, and M. Barlaud, "Motion-compensation lifted wavelet video coding: toward optimal motion/transform configuration," in *Proc. of European Signal Process Conference (EUSIPCO) 2004*, pp. 1975 – 1978, Sep. 2004.

[89] H. Brusewitz, " Motion compensation with triangles," in *Proc. of International Conference on 64kbits Coding of Moving Video*, Netherlands, Sep. 1990.

[90] G. Sullivan and R. Baker, "Motion compensation for video compression using control grid interpolation," in *Proc. IEEE International Conference of Acoustics, Speech and Signal Processing (ICASSP) 1991*, Canada, May 1991.

[91] C. Toklu, A. Erdem, M. Sezan, and A. Tekalp, "Tracking motion and intensity variations using hierarchical 2D mesh modelling for synthetic object transfiguration," *Graphical Models and Image Processing*, vol. 58, no. 6, pp. 553-573, Nov. 1996.

[92] P. Van Beek, A. Tekalp, N. Zhuang, I. Celasun, and M. Xia, "Hierarchical 2D mesh representation, tracking and compression for object-based video," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 9, no. 2, pp. 353-369, Mar 1999.

[93] G. Al-Regid, Y. Altunbasak, and R. Mersereau, "Hierarchical motion estimation with content-based meshes," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 13, no. 10, pp. 1000-1005, Oct 2003.

[94] Y. Wang and O. Lee, "Active mesh – a feature seeking and tracking image sequence representation scheme," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 3, no. 5, pp. 610-624, Sep. 1994.

[95] Y. Altunbasak and A. Tekalp, "Close-form connectivity-preserving solutions for motion compensation using 2-D meshes," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 6, no. 9, pp. 1255-1269, Sep. 1997.

[96] R. Szeliski and H. Shum, "Motion estimation with quadtree splines," *Technical Report, 95/1,* Digital Equipment Corp., Cambridge Research Lab, Mar 1995.

[97] C. Huang and C. Hsu, "A new motion compensation method for image sequence coding using hierarchical grid interpolation," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 4, no. 1, pp. 42-52, Feb. 1994.

[98] S. Cui, Y. Wang, and J. Fowler, "Mesh-based motion estimation and compensation in the wavelet domain using a redundant transform," in *Proc. IEEE International Conference on Image Processing (ICIP) 2002*, vol. 1, pp. 693-696, Sep. 2002.

[99] I. Richardson and Y. Zhao, "Video encoder complexity reduction by estimating skip mode distortion," in *Proc. IEEE International Conference on Image Processing (ICIP) 2004*, vol. 1, pp. 103-106, Sep. 2004.

[100] I. Ahmad, W. Zheng, L. Luo, and M. Liou, "A fast adaptive motion estimation algorithm," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 16, no. 3, pp. 420-439, Mar. 2006.

[101] M. Dudon, O. Avaro, and G. Eude, "Object-oriented motion estimation ," in *Proc. of Picture Coding Symposium*, pp. 284-287, Sep. 1994.

[102] J. Nieweglowski, T. Campbell, and P. Haavisto, "A novel video coding scheme based on temporal prediction using digital image wrapping," *IEEE Transaction on Consumer Electron*, vol. 39, pp. 141-150, Aug. 1993.

[103] S. Golomb, "Run-length encodings," *IEEE Transaction on Information Theory*, vol. 12, no. 3, pp.399-401, 1966.

[104] D. Taubman and M. Marcellin, JPEG2000: Image compression fundamentals, standards, and practice, Kluwer Academic Publishers, third edition, 2004.

[105] A. Yu, H. Park, and G. Martin, "Fast Mesh-based Motion Estimation Employing an Embedded Block Model," in *Proc. of IEEE International Symposium on Circuit and Systems (ISCAS) 06*, 4pp., Island of Kos, Greece, May 2006.

[106] A. Yu, H. Park, and G. Martin, "A Dual-Layer Approach for Improved Mesh-based Motion Compensation," under review by *IEE Proceedings Vision, Image, and Signal Processing,*, 10pp.

[107] A. Yu, H. Park, and G. Martin, "Rate-distortion Performance of the Dual-layer Motion Compensation Employing Different Mesh Densities," in *Proc. of 14th European Signal Processing Conference (EUSIPCO) 2006*, 5pp., Florence, Italy, Sep. 2006.

Publications

[1] Andy C. Yu, Ngan King Ngi, and Graham Martin, "Efficient Intra- and Inter-mode Selection Algorithms for H.264/AVC," in *Journal of Visual Communication and Image Representation – Special Issue on Emerging H.264/AVC Video Coding Standard*, vol. 17, issue 2, pp. 322-343, Elsevier Press, Apr 2006.

[2] Heechan Park, Andy C. Yu, and Graham Martin, "Progressive Mesh-based Motion Estimation", in *Lecture Notes in Computer Science (LNCS) – Visual Content Processing and Representation*, vol. 3893, pp. 98-101, Springer Press, April 2006.

[3] Andy C. Yu, Heechan Park, and Graham Martin, "A Dual-Layer Approach for Improved Mesh-based Motion Compensation," accepted for publication in *IET Proceedings of Image Processing*, 11pp., Mar. 2007.

[4] Andy C. Yu, Graham Martin, and Heechan Park, "Fast Inter-mode Selection in the H.264/AVC Standard Using Hierarchical Decision Process," accepted for publication in *IEEE Transaction on Circuits and Systems for Video Technology*, 10pp., Mar. 2007.

[5] Andy C. Yu, Heechan Park, and Graham Martin, "Rate-distortion Performance of the Dual-layer Motion Compensation Employing Different Mesh Densities," in *Proc. of 14th European Signal Processing Conference (EUSIPCO) 2006*, 5pp., Florence, Italy, Sep. 2006.

[6] Andy C. Yu, Heechan Park, and Graham Martin, "Fast Mesh-based Motion Estimation Employing an Embedded Block Model," in *Proc. of IEEE International Symposium on Circuit and Systems (ISCAS) 06*, 4pp., Island of Kos, Greece, May 2006.

[7] Andy C. Yu, Graham Martin, and Heechan Park, "Improved Schemes for Inter-frame Coding in the H.264/AVC Standard," in *Proc. of 12th IEEE Conference on Image Processing (ICIP) 05*, vol. 2., pp. 902-905, Genoa, Italy, Sep. 2005.

[8] Andy C. Yu, Graham Martin, and Heechan Park, "A Frequency Domain Approach to Intra Mode Selection in H.264/AVC," in *Proc. of 13th European Signal Processing Conference (EUSIPCO) 05*, 4pp., Antalya, Turkey, Sep. 2005

[9] Heechan Park, Andy C. Yu, and Graham Martin "Progressive Mesh-based Motion Estimation Using Partial Refinement," in *Proc. of 9th International Workshop on Very Low Bit-rate Video (VLBV) 05*, 4pp., Cagliari, Italy, Sep. 2005.

[10] Andy C. Yu and Graham Martin, "Advanced Block Size Selection Algorithm for INTER frame Coding in H.264/AVC," in *Proc. of 11th IEEE International Conference on Image Processing (ICIP) 04*, vol. 1, pp.95-98, Singapore, Oct 2004.

[11] Andy C. Yu, "Efficient Block Size Selection Algorithm for INTER frame Coding in H.264/AVC," in *Proc. of 29th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 04*, vol. 3, pp. 169-172, Montreal, Canada, May 2004.

[12] Andy C. Yu, "Fast algorithm for H.264/AVC mode selection," CS-RR-404, research report for Department of Computer Science, University of Warwick, United Kingdom, 2004.

[13] Andy C. Yu, "Improved schemes for multi-mode coding in the H.264/AVC standard and simplified Lagrangian evaluation for video coding," CS-RR-423, research report for Department of Computer Science, University of Warwick, United Kingdom, 2005.